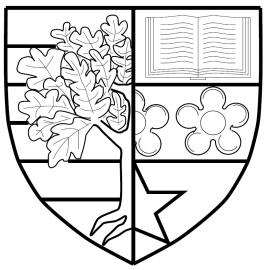
NOVEL METHODS FOR MULTI-TARGET TRACKING WITH APPLICATIONS IN SENSOR REGISTRATION AND FUSION

by
David Robb Cormack



A thesis submitted for the degree of
Doctor of Philosophy

School of Engineering and Physical Sciences
Heriot-Watt University, Edinburgh

May 2021

Abstract

Maintaining surveillance over vast volumes of space is an increasingly important capability for the defence industry. A clearer and more accurate picture of a surveillance region could be obtained through sensor fusion between a network of sensors. However, this accurate picture is dependent on the sensor registration being resolved. Any inaccuracies in sensor location or orientation can manifest themselves into the sensor measurements that are used in the fusion process, and lead to poor target tracking performance. Solutions previously proposed in the literature for the sensor registration problem have been based on a number of assumptions that do not always hold in practice, such as having a synchronous network and having small, static registration errors. This thesis will propose a number of solutions to resolving the sensor registration and sensor fusion problems jointly in an efficient manner. The assumptions made in previous works will be loosened or removed, making the solutions more applicable to problems that we are likely to see in practice. The proposed methods will be applied to both simulated data, and a segment of data taken from a live trial in the field.

Dedication

In loving memory of Robert and William Cormack.

Acknowledgements

There have been many ups and downs over the course of this work, and there is a very long list of people that deserve thanks, a lot more than I can fit on this page! Firstly, thanks to James Hopgood for taking over when times were very difficult, and to Yvan Petillot for his ongoing moral and personal support; and the main reason why I didn't throw in the towel a long time ago. A special thanks should go to Isabel Schlangen for her moral support from afar, her proofreading of this thesis, and for making the lab at Heriot-Watt a much happier place!

A big thanks should go to Leonardo in Edinburgh for the part-funding of this PhD, specifically to David Greig for setting up the project and making sure I never gave up; to Anthony Swain and Billy Spratt for their technical insights, knowledge, and support throughout this work; and to Colette Wilson and Lisa Blair for just being all round great pals.

I'm very lucky to have had a number of sporting outlets during the toughest times in the last few years. Thanks to the Monday night footballers (apologies for my competitiveness!); and the E=MCC cricketers, who have faithfully put up with my very questionable bowling for too long. Hopefully now I'll have a lot more free time to get some practice! To Mike and Doug, and the wider Hawk-Eye community, thanks for keeping me involved in rugby. To the medical teams at Edinburgh, Glasgow, and Scotland, I hope I've been more of a help than a hindrance... There should be some special mentions, firstly for Sam Hewitt; I will never tire of helping you find the right angle or saving a clip. Also, to Dave Pugh, thank you for your words of advice and support, usually coupled with some of the worst jokes known to man. And to Mike Proudfoot, thanks for the endless supply of tea and famous shortbread, and for having the guts to keep sticking swabs up my nose for the greater good. I don't think you all comprehend just how much you have done for me and lifted my spirits on many an occasion.

Lastly, but most importantly, to my partner Catherine. Words cannot describe how supportive you've been, and making sure I see this through to the very end. Without your support, I'm sure this thesis would never have happened. Thank you.

Contents

List of Tables

List of Figures

Acronyms

ANEES	Averaged Normalised Estimation Error Squared
BP	Belief Propagation
CFAR	Constant False Alarm Rate
CKF	Cubature Kalman Filter
COTS	commercial-off-the-shelf
CPHD	Cardinalized PHD
CPU	Central Processing Unit
CT	Coordinated Turn
DA	data association
Dstl	Defence Science and Technology Laboratory
EKF	Extended Kalman Filter
EM	Expectation-Maximisation
FC	Fusion Centre
FISST	Finite Set Statistics
FoR	frame of reference
FoV	Field-of-View
GLMB	Generalised Labelled Multi-Bernoulli
GM	Gaussian Mixture
GNN	Global Nearest Neighbour
GOSPA	Generalised Optimal Subpattern Assignment
GPB	Generalised Pseudo-Bayesian
GPS	Global Positioning System
GPU	Graphics Processing Unit
HLDMM	High-Level Decision Making Module
i.i.d.	independent and identically distributed
IMM	Interacting Multiple Model
IMU	Inertial Measurement Unit
IRST	Infrared Search and Track
JIPDA	Joint Integrated Probabilistic Data Association
JPDA	Joint Probabilistic Data Association

MAP	Maximum A Posteriori
MC	Monte Carlo
MCMC	Markov Chain Monte-Carlo
MGF	Moment-Generating Function
MHT	Multiple Hypothesis Tracking
MOL	Multi-Object Likelihood
MP	Message Passing
MTT	Multi-Target Tracking
NCA	near-constant acceleration
NCV	near-constant velocity
OMAT	Optimal Mass Transfer
OOSM	Out-of-Sequence Measurement
OSPA	Optimal Subpattern Assignment
PDA	Probabilistic Data Association
pdf	probability density function
PGF	Probability Generating Function
PGFL	Probability Generating Functional
PHD	Probability Hypothesis Density
PMF	Probability Mass Function
RB	Rao-Blackwellised
RCS	Radar Cross Section
RFS	Random Finite Set
RMSE	Root Mean Square Error
SAPIENT	Sensing for Asset Protection with Integrated Electronic Networked Technology
SAR	Synthetic Aperture Radar
SLAM	Simultaneous Localisation and Mapping
SMC	Sequential Monte-Carlo
SNR	Signal-to-Noise Ratio
SPA	Sum-Product Algorithm
SPADA	Sum-Product Algorithm for Data Association
STT	Single-Target Tracking

SWaP Size Weight and Power

UAV Unmanned Aerial Vehicle

UKF Unscented Kalman Filter

Notation

k	time/iteration index
n	target index
s	sensor index
$q \in \mathbf{q}$	registration parameter(s)
u	acceleration noise
$\mathbf{x} \in \mathbf{X}$	target state(s)
$\mathbf{z} \in \mathbf{Z}$	sensor measurement(s)
m	sensor measurement index
i	sample index
$f_{k\|k-1}(\cdot\|\cdot)$	state transition model
$l_k(\cdot\|\cdot)$	likelihood function
$h(\cdot)$	observation function
$\mathbf{F}$	state transition matrix
Δ_k	transition time
$\mathbf{w}$	process noise
ω	turn rate
$\mathbf{Q}$	process noise covariance
$\mathbf{H}$	observation matrix
η	measurement noise
$\mathbf{R}$	measurement noise covariance
$\mathbf{S}$	innovation covariance
$\mathbf{K}$	Kalman gain
$\mathcal{N}$	normal distribution
δ	Dirac delta function

w	particle weight
p_{d}	probability of target detection
p_{s}	probability of target survival
λ	average false alarm rate
ρ	range
ϕ	azimuth
$\hat{\ell}$	multi-object likelihood
$\mathbf{J}$	Jacobian matrix
r	target existence indicator
$\mathbf{y} = \{\mathbf{x}, r\}$	augmented state vector
$a \in \mathbf{a}$	target-oriented association variables
$b \in \mathbf{b}$	measurement-oriented association variables
$s_\star$	spatial distribution of false alarms
$\Psi(\mathbf{a}, \mathbf{b})$	exclusion enforcing function for association
$\tilde{f}(\mathbf{x}, r)$	belief approximation
ψ	factor

Chapter 1

Introduction

1.1 Motivations

The work that is presented in this thesis lends itself to the target tracking and sensor fusion areas. These areas have received a lot of attention in the signal processing literature, however there is one problem which is often overlooked, or dismissed, but is critical from a practical perspective: *the sensor registration problem* [1]. The use of large and sophisticated sensor networks to maintain coverage over wide geographical areas is becoming more common. The demand being placed on these networks is ever growing, as we could be interested in tracking many tens, or even hundreds of targets simultaneously [2], and these targets may be difficult to detect.

Consider a use case of drone detection and tracking [3], an application that is becoming more important after gaining a lot of negative press [4–10]. Imagine that we wish to protect an important asset or location such as an airport. Drones can often be flown in large swarms, and in tight formations, making the detection and tracking problems more difficult. Small Unmanned Aerial Vehicles (UAVs) tend to be made of lightweight and low-reflectivity materials such as plastic and carbon fibre, further increasing the difficulty of the detection problem, and hence the need for deploying *heterogeneous sensors*. The use of sensor fusion allows for the exploitation of more sensor information and would result in a clearer, more accurate tracking picture [11] *assuming that the sensors are registered correctly and biases are accounted for or removed.* Having incorrect registration will lead to the formation of inaccurate tracks through biased measurements; a major issue for real-world systems [1]. The simple scenario presented in Figure 1.1 highlights the effect that these errors can have, just from a small angular bias between the sensors. Suppose that we have a static platform with two sensors attached, and we wish to detect and track a target that is at a range of 5 km. By simple trigonometry, if there was an angular registration error of 2° between the two sensors, one of the two sensors would measure the target approximately 175 m away from it's true location; a value that would only get larger if we were interested in detecting targets further

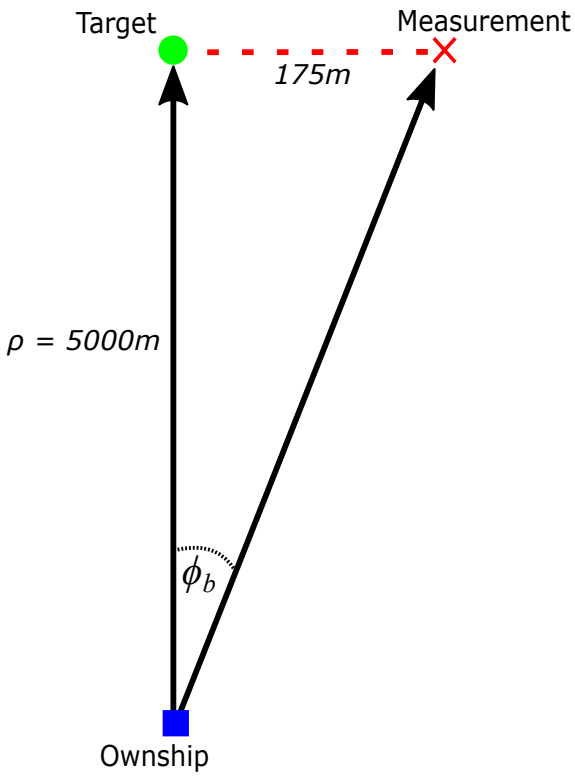


Figure 1.1: An example of the registration problem. If $\phi_b = 2°$, the measurement would place the target approximately 175 m away from its true location.

away from the platform.

Previous solutions that have been proposed for correcting the sensor registration have been developed on a range of assumptions, that often do not hold true in a range of real scenarios of interest in this application area. Some examples of these problems include:

Sensors operating synchronously and all providing measurements at a given instance in time [12–14]. Given the commercial-off-the-shelf (COTS) nature of many sensors, which may then be used in an ad-hoc sensor network or fusion system, it is highly unlikely that the synchronous assumption holds.

Offline registration would give sensors an initial set of registration parameters to work with. Systems are often calibrated when they are installed to a new platform, and when first powered on, however, it is likely that these initial conditions will change over time through factors such as sensor drift or platform vibrations. It cannot be assumed that *the calibration parameters will remain the same over the course of a scenario* [12–16].

Registration to a global frame of reference (FoR) such as the World Geodetic System, may not always be possible, especially in Global Positioning System (GPS)-denied environments. It should be noted that finding this global calibration, or registration, is typically a very difficult problem in practice.

Homogeneous sensors are a feature of more traditional sensor networks [12–17], however this is becoming less common in modern systems as more data can often be gleaned from heterogeneous sources.

Very small relative biases have been assumed and presented in the previous sensor registration literature which seem unrealistic compared to those seen in practice. For example in [13, 17], the assumed angular biases are in the order of tenths of degrees. Angle biases an order of magnitude larger than this are addressed in this thesis.

The sensor registration and fusion schemes developed in this thesis will provide solutions that resolve the issues listed above. These schemes must allow for the online estimation of registration parameters, and perform in an efficient manner, such that they could be deployed to real systems.

1.1.1 Multi-Target Tracking and Fusion

When considering highly dynamic scenarios, the number of targets, and the states of those targets, will vary over time due to targets entering and leaving the surveillance region. After many advances in the aerospace area during the 1960s, Multi-Target Tracking (MTT) became a much more prominent and important problem that needed to be resolved in practice. Advances in the MTT algorithms, alongside refinement in the sensors supplying measurements to them, have allowed for more application areas to embrace the MTT paradigm [18–21].

MTT algorithms need to be able to contend with many other aspects that are not typically found when considering the Single-Target Tracking (STT) problem, such as false alarms, the data association (DA) problem, and missed detections. Each algorithm has its own approximation or formulation to handle these issues. The algorithms are broadly split into two categories; point process methods and vector-type methods, which will be discussed in more depth in Section 2.3. Both the point process methods and vector-type methods will be exploited later in this thesis. The aspects of the algorithms that are of most interest to this thesis will be the trade-offs between performance and computational efficiency. With surveillance scenarios potentially changing quickly over time, it is important that the algorithms can process measurements quickly and provide suitably accurate outputs. Other important factors that need to be considered are the scalability of the algorithm in the number of targets (this has been addressed and discussed in the previous literature e.g. [19,22]), and the scalability in the number of sensors [22,23]. The latter has been partially addressed in previous works, but not in situations or scenarios where the sensors are not accurately calibrated.

With the large advancements in multi-sensor suites in recent years, being able to track multiple targets using a range of different sensing modalities is now feasible

[24–28]. Sensor fusion is now a well-developed methodology that aims to automate the process of combining multiple different sources of information [11, 29, 30]. By combining these multiple sources of information, the output should in some sense, be better than if the multiple sources of information were each used individually. In this case, "better" could mean:

- the improvement of the target tracking update rate;
- reducing the spatial uncertainty in a target's estimated position;
- reducing the spawning of false or "ghost" tracks; and
- maintaining a higher level of all-round situational awareness.

Because practical implementations of tracking and fusion algorithms are typically built on a number of approximations, it may be the case that these benefits are not fully achieved, or will be difficult to observe.

1.1.2 Sensor Registration

As will be highlighted throughout this thesis, the sensor registration problem is fundamental to any type of sensor fusion system. Sensor registration, also referred to as sensor calibration in the literature, involves estimating sensor parameters such as location and orientation alongside the multiple target states. Calibrating MTT systems [31] through the estimation of appropriate model and system parameters is an important validation and verification step before a system is deployed out in the field. When attempting to fuse measurements from multiple different sensors, the registration on to a common frame of reference (FoR) must be considered, or we risk reducing the overall performance of the sensor fusion and the underlying tracking algorithm. Sensor registration errors, or biases in sensors, can appear in the system from many different sources:

Human errors could account for problems such as the the incorrect installation of sensors on to a platform. Some lower-cost COTS sensors may not perform their own automated calibration routine during the power-on step, and parameters such as the limits of the Field-of-View (FoV), and the sensor's orientation angles, may need to be preset by an engineer or operator. If these values are not computed correctly, or not input correctly, the sensor may think it is looking in one direction, whereas in reality it is pointing in a different direction. Another issue that could arise is the intrinsic design, and calibration, of smaller subsystems inside the larger sensor. Once these smaller components are manufactured and installed into the system, they are unlikely to be recalibrated or adapted unless absolutely necessary. These intrinsic issues could then propagate through the sensor and create larger system errors.

Sensor errors could appear due to measurements captured in an external system. Many platforms are fitted with GPS and Inertial Measurement Units (IMUs) to gain an accurate location and orientation at any given time. However, these systems are likely to drift over time and accumulate a larger error [32]. If this type of error is not continually accounted for, the sensor fusion performance is likely to slowly decrease over time. The GPS performance and accuracy is likely to vary over time, depending on how many satellites the receiver can see at a given time. This error is more commonly referred to in the navigation literature as the dilution of precision [33]. A similar problem to this, often found in the robotics and navigation literature, is the problem of Simultaneous Localisation and Mapping (SLAM). The SLAM problem is quite similar to the sensor registration problem being resolved in this thesis; however their focuses are a little different. In sensor registration, our main focus is to estimate the states of targets, or landmarks, with less of a concern about the navigation aspect. Conversely, in the SLAM problem, the main focus is on estimating appropriate navigation states and parameters such as a local map. There is much less emphasis on the particular landmarks being used in the SLAM problem.

Network errors could have a major effect on the ability to perform data fusion in a practical sense, especially in situations where the sensors are located a long distance away from one another. Any *latency or timing issues* would need to be known and accounted for in the system, so that sets of measurement data do not appear at the Fusion Centre (FC) in the wrong order. Attempting to use data that is not properly time-aligned could lead to poor tracking performance [21]. This *timing registration issue* highlights the need for a common clock that sensors should take their timing information from. The amount of communication bandwidth available on either a physical, or a wireless link, could affect the number of measurements or tracks that can be transmitted to the FC at any one time. This again may cause issues with certain, or all, tracks being updated in a timely manner. Furthermore, there may be cases in wireless systems where the connection to the FC is not available for any number of reasons. These issues and errors can make the real-world application of fusion much more difficult.

Weather and external forces are a more difficult factor to predict and account for in practice. They are inherently random and could occur at any point in time. Consider a first example of a large warship in rough seas. One set of sensors could be located at the bow of the ship and another set of sensors at the stern. Large waves could cause flex in the ship's structure, causing the registration between the two sets of sensors to change considerably over time. If one of the sets of sensors was on top of a large mast, this problem could be

exacerbated by strong winds and further increase the registration error. One further application in the maritime surveillance area could be for sonobuoys which are dropped onto the sea surface. In rough seas, they are likely to move around continuously and their state will change, making a continuous registration scheme necessary. Now, consider an example for the airborne application, where we could have a single set of sensors on a fast jet. With the platform potentially flying at supersonic velocities, there are extreme forces being applied to the platform. If sensors and their mounts to the aircraft are not strongly attached, it is possible that the sensors will physically shift during flight, and hence the need for a continual calibration strategy.

With the potential for these errors to be time-varying, online sensor registration schemes [34] are a necessity in practical applications. There may also be extreme cases where it may be impossible to calibrate the system to any sort of global reference frame e.g. in a GPS-denied environment [35].

With the high importance of the registration problem in practice it is concerning to see that, in a lot of the sensor fusion literature, the problem is either resolved separately to the tracking problem or is avoided altogether; a worrying trend for those interested in practical application and deployment to a system. If the problem is treated separately, there is still a chance of residual, or systematic, biases not being overcome. Another issue that is not covered by many articles in the literature is that of asynchronous sensors. Many works assume that the sensors operate synchronously and measurements are available for all sensors at a given iteration; this is very unlikely to happen in practice. Real-world radar systems may be designed to perform a number of different tasks depending on user selection, or through a form of automated sensor management, at any given point in time. These tasks may include:

- performing a full sweep/search of the surveillance region;
- interrogating a specific area inside the region to gain more measurement data, potentially to gain more information about a specific target for example; or
- creating a Synthetic Aperture Radar (SAR) image [36] of a specific area.

Each of these tasks will take a varying amount of time to complete, and not all tasks will provide a set of measurements that we can exploit to update the target tracks that we have. This forms a part of the larger asynchronous sensing problem; in practice *we do not know exactly when the next set of measurements will be available to the tracker*. This also holds true in the multi-sensor tracking case; we need accurate timing information so that the target dynamic model (see Chapter 2) is applied correctly. One further issue that arises in real-world systems is the need for efficient algorithms, which can be run in real-time on a platform with a limited amount of

processing power. Many pieces of literature propose solutions that may be effective for ground-based systems which may hold a lot of computing power in reserve should it be required; however with airborne applications, any high complexities will either need to be approximated, or completely avoided.

There have been a number of attempts to resolve the registration problem using machine learning and neural network solutions [37, 38], however this also comes with a number of disadvantages. These methods can require a very large amount of realistic training data to give sensible results, which can be difficult to generate, or record from a real scenario. There are also issues with "explainability" when considering machine learning techniques; very limited output information is given to *explain how* a machine learning algorithm reached a particular decision [39, 40]. Alternatively, another popular technique for sensor registration is to use pseudo-measurement approaches [13, 15, 16], where all of the biases to be estimated are placed in to a stacked vector with the target states. This technique often reduces the problem down to only pairs of sensors. These will be expanded upon in Section 2.5.

1.2 Scope of Thesis

While the MTT and data fusion problems have had widespread literature coverage, the sensor registration problem has had much less research focus. As emphasised in Section 1.1.1, efficient MTT algorithms are required to process data quickly for time-sensitive application areas such as defence and surveillance. This efficiency will provide the key starting point to develop the registration and fusion methods discussed in Section 1.1.2.

1.2.1 Research Questions

With this overall scope in mind, this thesis will attempt to address the following research questions:

1. When resolving the registration and fusion problems, what are the main considerations that should be taken into account when choosing an appropriate framework or algorithm?

2. How can a method be made scalable to a larger number of sensors? This could be a key requirement for a real-world system. Having a larger number of sensors available on a single platform, or across multiple platforms in practice could be important to improve the probability of detecting a target and to build sensor redundancy into the fusion system. There is also the argument that you could use a large number of low Size Weight and Power (SWaP) and low-cost sensors, rather than using one high-performance sensor to perform the

same tasks. This would of course however bring in a much larger registration problem to be resolved; there is always a trade-off.

1.2.2 Use Cases

The high-level objective of this thesis is to investigate sensor registration and fusion frameworks that could be applied to problems in the defence industry. Two main use cases will be used throughout this work; firstly multistatic radar (Figure 1.2), and secondly heterogeneous sensing (Figure 1.3).

Multistatic Radar

By exploiting multistatic radar networks, a number of distinct advantages are available over the traditional single-radar operation. The first of these is the improved ability to discriminate targets that may be close to one another. Depending on the geometry of the scenario and the locations of the radars, having a different look aspect to the targets may show a larger Radar Cross Section (RCS) [36, 41] and therefore improve the radar detection. A varying RCS example is shown in publication C1 for a typical drone that can be bought off-the-shelf.

As shown in Figure 1.2, by carefully choosing the geometry of the radars, it may be possible to vastly reduce the uncertainty in a target's location. Modern radar systems typically have a very high accuracy when creating a down-range measurement by rapidly sampling the received signal. Because of the high sampling rate, the down-range measurement error is likely to be in the order of a few metres. However, the cross-range uncertainty in the measurement is likely to be much larger than this. Without a range of advanced signal processing techniques such as monopulse [36, 42], we can only say that the detected target is inside the beam or not. If we consider larger radar beam-widths of 6° or more, this could translate to distances of tens or hundreds of metres of uncertainty on the ground. Effectively, we could use the down-range measurement in one of the radars to correct for the cross-range measurement in the other radar, and vice-versa [43, 44].

There are other potential advantages of using multistatic radar in practice:

Micro-Doppler effects can aid the identification and classification of targets. Each target will produce a Doppler shift depending on its velocity with respect to the radar. If the target were to contain moving or rotating parts (e.g. wheels, blades, rotors etc.), it may be possible to detect the micro-Doppler effect [45, 46]. Having more aspects of the target visible to the multiple radars at a given time may allow for more certainty when classifying a target.

Clutter reduction may be possible, but again is dependent on the sensor geometry. Consider a sea clutter example where rough sea swells directed towards the radar may create a number of false alarms [47, 48]. By having another

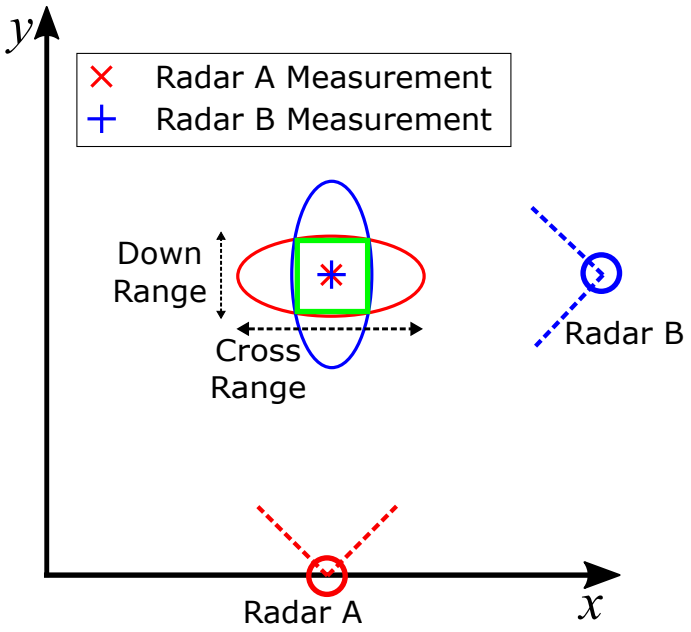


Figure 1.2: Multistatic radar use case.

radar at a different location, and therefore having a different look angle, the number of clutter points seen could be vastly reduced.

Survivability could also be seen as an advantage in this case (although it could potentially apply to other multi-sensor systems). If a sensor fault occurred in the single-radar case, the whole system would be offline. However for multi-sensor cases, only the one node would be offline and information from other sensors could still be exploited. If we were to consider the possibility of interference from external sources, having multistatic radar would be a key advantage; it would be very difficult to focus the interference in such a way that all radars would be affected simultaneously [44].

Heterogeneous Sensing

The inclusion of heterogeneous sensors, rather than using just one sensor modality across the sensor network (as with multistatic radar), could make it easier to detect certain types of target. Relating back to the UAV example, using only radar could make detection of small targets or targets with small RCS [36, 41] very difficult, especially at longer distances; however the use of optical or infrared sensors could aid detection as bright spots such as the battery pack will be easier to see. This will aid the detection task, but could also lead to improvements when looking at the higher-level identification and classification tasks. Features found in the differing types of sensor could again be combined to gain more certainty on the type of target.

Having different types of sensor may also bring different levels of accuracy and resolution to the larger sensor fusion picture. Consider a situation with a colocated

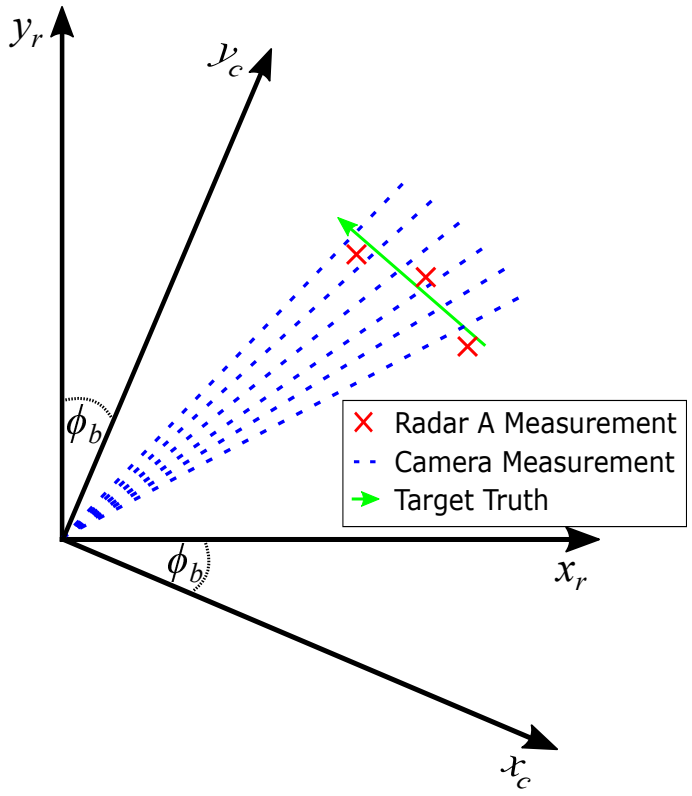


Figure 1.3: Heterogeneous sensing use case.

radar and an imaging system attached to a common platform, and recall the uncertainty reduction explanation above. For that to apply in multistatic radar, it is necessary to have sensors that are physically separated and exploit the overlap in uncertainty. For the colocated system, this is not required. The angular resolution of the imaging system is likely to be much higher than the angular resolution of the radar; hence the uncertainty reduction is directly possible.

One further prominent advantage of using an imaging system in the sensor network is the large increase in the measurement rate. A radar may take a number of seconds to perform a sweep of the surveillance region, or to decide to return and look along a particular sightline, and create the next set of measurements to be used in the tracker. An imaging system could take a fraction of that time to form an image of the region and pass measurements to the tracker. With all of this data available, target tracks can be maintained at a much higher rate, giving better all-round situational awareness.

1.2.3 Organisation

This thesis is organised into the following chapters:

Chapter 2 introduces typical Bayesian filtering for estimation and tracking problems. A number of common STT algorithms and MTT algorithms will be introduced as a precursor to their exploitation in later chapters; in order to

put the proposed methods in to context, a number of state-of-the-art methods in sensor registration are presented.

Chapter 3 provides an overview of the modelling assumptions that the algorithms presented in later chapters are based on. This chapter will also present the chosen fusion architecture that will be exploited throughout.

Chapter 4 presents the initial concept and design of a robust joint registration and fusion framework, based on *single-cluster methods*, taken from the Random Finite Set (RFS) literature. The framework contains an Extended Kalman Filter (EKF)-Probability Hypothesis Density (PHD) filter to perform the MTT, and a grid-based method [49] to estimate the registration parameter. Results are provided on the heterogeneous sensing scenario on both simulated data and a portion of real data. There are some issues with this implementation, such as the inability to maintain a unique track history for each individual target and the restrictiveness of the grid-based method, which leads on to Chapter 5 where a number of these are resolved.

Chapter 5 makes advancements to the method proposed in Chapter 4 by replacing the EKF-PHD filter with a Message Passing (MP) method for MTT, which leads to a number of advantages, such as the ability to maintain track identity for example. In order to implement this, a suitable Multi-Object Likelihood (MOL) (Section 2.3.3) has been derived to link the algorithm to the particle filter used in the registration estimation. The use of the particle filter overcomes the dimensionality issues when attempting to scale the grid-based method to multi-parameter estimation. Results using simulated data are shown for both the multistatic radar and heterogeneous sensing scenarios.

Chapter 6 looks at the problem of registration and fusion in networks containing larger numbers of sensors. Now that we have the desired robust implementation from Chapter 5 that is suitable for the 2-sensor case, further simulations have been carried out to test the scalability of the algorithm with respect to the number of sensors, and the number of particles used to estimate the registration parameter(s). A very recent addition to the literature [50] provides a suitable comparison for the results generated in this chapter. The simulated results presented are based on a multistatic radar scenario, and show that the proposed method from Chapter 5 *achieves very similar accuracy* to that of the method in [50]; but requires *a longer execution time* to reach this result.

Chapter 7 provides a summary of the work carried out in this thesis, and provides suggestions of where this research could be continued in future.

1.2.4 Contributions

This thesis proposes new solutions to resolving the registration and fusion problems, which are of significant practical importance. The key contributions and highlights of this thesis are:

1. the development of scalable sensor registration algorithms such that they could handle real-world issues such as resource-constrained platforms, dynamic time-varying sensor biases, and asynchronous sensors (Chapter 4, Chapter 5 and Chapter 6).

2. the further development of the *single-cluster method*, such that it can incorporate MTT and sensor calibration in a joint manner for sensor fusion systems - a new application area for this method (Chapter 4).

3. a novel extension to MP for MTT algorithms, such that they can be incorporated in to the hierarchical framework; this includes the derivation of a suitable new MOL function (Chapter 5).

4. a comprehensive range of simulations, based on varying types and sizes of sensor setup, and with challenging target scenarios including crossing targets; moreover, the simulations consider much larger biases (by at least one order of magnitude) compared to those presented in previous literature (Chapter 4, Chapter 5 and Chapter 6).

1.2.5 Publications

The following articles have been published as a result of the work presented in this thesis. Article J1 is based on the work shown in Chapter 4, and articles J2, C2 and C3 are based on the work in Chapter 5.

J1: D. Cormack, I. Schlangen, J. R. Hopgood and D. E. Clark, "Joint Registration and Fusion of an Infra-Red Camera and Scanning Radar in a Maritime Context," in *IEEE Transactions on Aerospace and Electronic Systems.* 2019.

J2: D. Cormack and J. R. Hopgood, "Message Passing and Hierarchical Models for Simultaneous Tracking and Registration," *accepted in: IEEE Transactions on Aerospace and Electronic Systems.* 2021, to appear.

C1: D. Cormack and D. Clark, "Tracking Small UAVs Using a Bernoulli Filter," in *Sensor Signal Processing for Defence (SSPD) 2016.* Edinburgh, UK: IEEE, 2016.

C2: D. Cormack and J. R. Hopgood, "Message Passing for Joint Registration and Tracking in Multistatic Radar," in *Sensor Signal Processing for Defence (SSPD) 2019.* Brighton, UK: IEEE, 2019.

C3: D. Cormack and J. R. Hopgood, "Sensor Registration and Tracking from Heterogeneous Sensors with Belief Propagation," in *22nd International Conference on Information Fusion (FUSION) 2019.* Ottawa, CA: IEEE, 2019.

Chapter 2

Background Theory and State-of-the-Art

The application of Bayesian methods for target tracking are fundamental to the outcomes in this thesis. Many methods presented in the academic literature are based on strict or unrealistic assumptions that often do not translate well when we wish to implement them in practice. This chapter will introduce a number of fundamental concepts in target tracking that are important for the work and solutions presented in later chapters, and also reviews recent literature in which they appear.

Firstly, Section 2.1 will introduce the two general underlying models that tracking algorithms are based on. Section 2.2 will introduce the Bayesian estimation problem, and explore some important Single-Target Tracking (STT) algorithms. The move from STT to the Multi-Target Tracking (MTT) problem is far from trivial; Section 2.3 will outline the key differences, and show the two main types of MTT algorithm that appear in the literature. In order to understand the Message Passing (MP) algorithms that are applied during Chapter 5 and Chapter 6, an introduction to factor graphs is provided in Section 2.4. The registration problem is a key aspect of the larger problem being resolved in this thesis and Section 2.5 will look at a range of recent solutions to sensor registration estimation and outline the current state-of-the-art in this area. Finally, Section 2.6 will define the metrics and mathematical tools that are used throughout this thesis to compare estimation performance between the different methods. Other pieces of more focused theory will be presented in each of the individual chapters when required.

2.1 Tracking Models

Bayesian target tracking algorithms are heavily based upon two underlying models; a *dynamic model* that represents the target(s) motion from one point in time to the next, and an *observation model* that represents the relationship between the sensor

measurements and the state-space coordinate system.

2.1.1 Dynamic Models

Dynamic models underpin the prediction step found in Bayesian tracking algorithms. The most popular form of dynamic models are *kinematic state models*, which are found by setting a certain derivative of the position to zero mean white noise. These models also include *process noise*, which models the uncertainty in the chosen dynamic model. Dynamic models could also describe other phenomena than target motion, such as temperature or financial stocks.

In practice, a model must be chosen based on a best guess of how we think a target will evolve over time. All of the models are based on the laws of physics. It is indeed possible that over time, a target may move according to one motion model, and then perform a manoeuvre that is a closer fit to an alternative model. Multiple model tracking is possible through the use of the Generalised Pseudo-Bayesian (GPB) approach, or potentially through an Interacting Multiple Model (IMM) structure [51].

Three of the most commonly used models for target motion are the near-constant velocity (NCV) model, near-constant acceleration (NCA) model and the Coordinated Turn (CT) model; these will be briefly introduced next. A much wider range of motion models are introduced and explained in [52].

Near-Constant Velocity model

The NCV model, which closely relates to the Discrete White Noise Acceleration Model, is a very popular model choice in practice. It is robust to small manoeuvres that last for short periods of time. The model itself can be obtained by discretising the continuous-time system, containing a state vector $\mathbf{x}$ (assumed to contain position and velocity components), and driven by a zero-mean white acceleration process noise. After discretisation, the discrete-time state equation at time k is

$$\mathbf{x}_{k|k-1} = \mathbf{F}_k \mathbf{x}_{k-1} + \mathbf{w}_k. \tag{2.1}$$

For a 1-dimensional problem, the transition matrix $\mathbf{F}_k$ is comprised of

$$\mathbf{F}_k = \begin{bmatrix} 1 & \Delta_k \\ 0 & 1 \end{bmatrix}, \tag{2.2}$$

where Δ_k is the transition time, and the process noise term $\mathbf{w}_k$ is assumed to be a zero-mean white sequence with variance σ_w^2. This term has covariance defined by

$$\mathbf{Q}_k = \sigma_w^2 \begin{bmatrix} \frac{1}{4}\Delta_k^4 & \frac{1}{2}\Delta_k^3 \\ \frac{1}{2}\Delta_k^3 & \Delta_k^2 \end{bmatrix}. \tag{2.3}$$

Near-Constant Acceleration model

From a practical perspective, a more applicable and easier to define model for accelerating targets is the piecewise constant Wiener process acceleration model. In practice, we may have a good idea of the expected target acceleration values, but we are much less likely to know reasonable values for the *jerk* parameter (the fourth order term). Because of this, the continuous time Wiener process acceleration model is more difficult to tune in the real world.

The discrete model has a third order state equation (now including position, velocity and acceleration values)

$$\mathbf{x}_{k|k-1} = \mathbf{F}_k \mathbf{x}_{k-1} + \mathbf{\Gamma}_k \mathbf{w}_k \tag{2.4}$$

with

$$\mathbf{F}_k = \begin{bmatrix} 1 & \Delta_k & \frac{1}{2}\Delta_k^2 \\ 0 & 1 & \Delta_k \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.5}$$

$$\mathbf{\Gamma}_k = \begin{bmatrix} \frac{1}{2}\Delta_k^2 \\ \Delta_k \\ 1 \end{bmatrix}, \tag{2.6}$$

and the process noise term $\mathbf{w}_k$ is the acceleration increment over a sampling period, assumed to be a zero-mean white sequence with variance σ_w^2. The process noise covariance term can be shown to be

$$\mathbf{Q} = \mathbf{\Gamma}_k \sigma_w^2 \mathbf{\Gamma}'_k = \sigma_w^2 \begin{bmatrix} \frac{1}{4}\Delta_k^4 & \frac{1}{2}\Delta_k^3 & \frac{1}{2}\Delta_k^2 \\ \frac{1}{2}\Delta_k^3 & \Delta_k^2 & \Delta_k \\ \frac{1}{2}\Delta_k^2 & \Delta_k & 1 \end{bmatrix}. \tag{2.7}$$

This NCA model is appropriate for situations where the acceleration is almost constant for a sustained period of time (say at least 10 tracker iterations) [21].

Coordinated Turn model

Most vehicular turning manoeuvres follow a Coordinated Turn (CT) model. This can be characterised with a nearly constant turn rate, and a nearly constant velocity. Although turns in the real world are not coordinated in the strictest sense (for an aircraft the ground speed is the air speed plus any wind speed), it can still be modelled by a CT model with an amount of noise representing the error.

The state vector contains positions, velocities and accelerations. The three-dimensional turn rate model is a non-linear model, as the velocity and acceleration estimates present in the state vector are used to compute the turning rate that is

used inside the state transition matrix. The transition matrix for the CT model is

$$\mathbf{F}_k = \begin{bmatrix} 1 & \omega_k^{-1}\sin(\omega_k\Delta_k) & \omega_k^{-2}(1-\cos(\omega_k\Delta_k)) \\ 0 & \cos(\omega_k\Delta_k) & \omega_k^{-1}\sin(\omega_k\Delta_k) \\ 0 & -\omega_k\sin(\omega_k\Delta_k) & \cos(\omega_k\Delta_k) \end{bmatrix}, \tag{2.8}$$

where ω_k is the turn rate computed from the target state vector, and the process noise covariance $\mathbf{Q}$ is the same as that of the NCA (see (2.7) above) [53].

2.1.2 Observation Models

The observation model helps form the update step of Bayesian tracking algorithms by defining the relationship between the target state vector and the measurements that are generated by the sensor. There are two main types of sensor that are found in practice, *active* sensors and *passive* sensors.

Active sensors such as radar and sonar send out energy (be that acoustic or electromagnetic) to try and detect targets. Reflections of the transmitted signal will be received by the sensor, allowing for position, and potentially velocity, measurements to be generated.

Passive sensors which could also be acoustic and electromagnetic, or potentially optical-based, do not emit any energy of their own. This means that they do not provide a full position measurement; only angular directions of arrival are available, without any range information. Consider the optical sensor case; each pixel in the created image could relate to an azimuth and elevation value in the real world through a three-dimensional projection [21]. Passive sensing is a very important capability for modern platforms as it can make the platform less likely to be detected by other sensors. By not emitting any energy, sensors, and the platform they are on, can stay hidden until absolutely necessary.

There are two main relationships found in observations models, depending on how the sensor measurements can be related to the state vectors. This may either be a linear, or a non-linear, relationship.

Linear Relationship

Having a linear relationship between the observation space and the state space, and linear dynamics, allows for an optimal solution to the tracking problem. An example of a linear relationship in practice could be the tracking of an object in an image. The state vector for the target(s) in the image may be in pixel coordinates (with position and velocity components), and the output of a detector that attempts to generate measurements from the image may provide pixel positions. From this, we can see that the position components of the state and measurement vectors are described in the same state space. In such a case, the observation matrix $\mathbf{H}$ is

constructed such that the appropriate variables from the state vector are selected for use in the residual calculations in the Kalman filter update step [54].

Non-linear Relationship

Many problems in practice cannot be constructed using a linear relationship, and non-linear alternatives are required. This is often the case in radar target tracking, where the state vector may be in Cartesian coordinates, and measurements are often generated in polar or spherical coordinates. This is a problem for the Kalman filter formulation, as this non-linear function does not preserve the *Gaussianity* of the target distribution. In this case, an *approximation* is required to ensure that the posterior distribution becomes Gaussian.

Some suitable approximations that can be made include a linearisation around the current state estimate (found in the Extended Kalman Filter (EKF)) or the use of a set of deterministic sigma points to perform the transformation between spaces (found in the Unscented Kalman Filter (UKF)). The EKF will be exploited to overcome the non-linear problem found in this work, and will be introduced in more detail in Section 2.2.1.

2.2 Single-Target Tracking

Target tracking can be seen as a state estimation problem, where the state(s) of the target(s) will vary over time. The models used to approximate the target dynamics are often described in discrete time; however there are a number of non-linear and continuous-time models available [52]. Often, the state transition is described using a first-order Markov model denoted $f_{k|k-1}(\mathbf{x}_k|\mathbf{x}_{k-1})$.

For the Single-Target Tracking (STT) formulation, at each time-step k, the target state $\mathbf{x}_k$ will generate a measurement $\mathbf{z}_k$ according to some measurement equation

$$\mathbf{z}_k = h_k(\mathbf{x}_k, \mathbf{v}_k), \tag{2.9}$$

where $h_k(\mathbf{x}_k, \mathbf{v}_k)$ is a (non)linear transformation of the state vector and $\mathbf{v}_k$ represents some additive measurement noise dependent on the chosen model. The measurement equation is often described using a likelihood function

$$l_k(\mathbf{x}_k|\mathbf{z}_k). \tag{2.10}$$

which describes the association of a measurement $\mathbf{z}_k$ to a state $\mathbf{x}_k$ at time k. As the target moves over time, all of the state history from the initial time-step through to time-step k is captured inside the posterior density $p_{0:k}(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})$ where $\mathbf{z}_{1:k}$ is the measurement history. The filtering density $p_k(\mathbf{x}_k|\mathbf{z}_{1:k})$ is a marginal of this posterior

density, which can be defined as the probability density of the state at time k, given the measurement history $\mathbf{z}_{1:k}$.

The initial density p_0 is often carefully chosen in practice (explored in Section 2.3) so that a target track can be initialised. From this initial density, a Bayes recursion containing the Chapman-Kolmogorov equation [55, 56] and the Bayes update is performed to reach the filtering density at time k using the equations

$$p_{k|k-1}(\mathbf{x}_k|\mathbf{z}_{1:k-1}) = \int f_{k|k-1}(\mathbf{x}_k|\mathbf{x}_{k-1}) p_{k-1}(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1}) \mathrm{d}\mathbf{x}_{k-1}, \tag{2.11}$$

$$p_k(\mathbf{x}_k|\mathbf{z}_k) = \frac{l_k(\mathbf{x}_k|\mathbf{z}_k) p_{k|k-1}(\mathbf{x}_k|\mathbf{z}_{1:k-1})}{\int l_k(\mathbf{x}'|\mathbf{z}_k) p_{k|k-1}(\mathbf{x}'|\mathbf{z}_{1:k-1}) \mathrm{d}\mathbf{x}'}. \tag{2.12}$$

These equations form the underlying recursion found in many tracking algorithms such as the Kalman filter.

2.2.1 Kalman Filter/Extended Kalman Filter

The Kalman filter [54] is a well-known method for performing state estimation in a linear system. It provides an optimal closed-form solution to the Bayes recursion if we consider the underlying models to be linear and Gaussian. Both the dynamical model and the measurement model are linear transformations with additive Gaussian noise such that

$$\mathbf{x}_k = \mathbf{F}_k \mathbf{x}_{k-1} + \mathbf{w}_k, \tag{2.13}$$

$$\mathbf{z}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k, \tag{2.14}$$

where $\mathbf{F}_k$ is the state transition matrix, $\mathbf{w}_k$ is the process noise term, $\mathbf{H}_k$ is the observation matrix, and $\mathbf{v}_k$ is the measurement noise (both dependent on the measurement model for the sensors being used). The process and measurement noises are generated through zero-mean Gaussian variables with covariance matrices $\mathbf{Q}_k$ and $\mathbf{R}_k$ respectively. These lead to a transition density of

$$f_{k|k-1}(\mathbf{x}_k|\mathbf{x}_{k-1}) = \mathcal{N}(\mathbf{x}_k; \mathbf{F}_k \mathbf{x}_{k-1}, \mathbf{Q}_k), \tag{2.15}$$

and the likelihood function

$$l_k(\mathbf{x}_k|\mathbf{z}_k) = \mathcal{N}(\mathbf{z}_k; \mathbf{H}_k \mathbf{x}_k, \mathbf{R}_k), \tag{2.16}$$

where $\mathcal{N}(\mathbf{z}; \mathbf{m}, \mathbf{P})$ denotes the distribution for a Gaussian random variable with mean $\mathbf{m}$ and covariance $\mathbf{P}$. The Kalman filter recursion equations are published in many other pieces of literature [19, 54, 57] and will not be presented here.

The Kalman filter is the optimal solution for state estimation when the modelling assumptions are all linear. However in practice, there are many applications which

do not follow these linear assumptions. For example in the scenarios considered in this thesis, the state-space is in Cartesian coordinates, and the sensor measurement space(s) is/are in Polar coordinates. In order to overcome this non-linearity between spaces, we need an alternative to the basic Kalman filter. Two suitable methods are presented here and exploited later, the first of which is the Extended Kalman Filter (EKF).

The EKF is a sub-optimal estimation algorithm which can handle non-linear models [58]. It uses a Taylor series expansion of the dynamic and measurement model up to the first-order term, and hence disregards all of the non-linear components. Rather than having the observation matrix $\mathbf{H}_k$ and being able to directly observe state variables in the received measurements, $\mathbf{H}_k$ is now a Jacobian matrix, made up of partial derivatives, such that

$$\mathbf{H}_k(\mathbf{x}) = \frac{\partial h}{\partial \mathbf{x}_k}, \tag{2.17}$$

where h is a non-linear function. The use of a series expansion in either the prediction or update steps introduces extra unmodeled error in to the estimation process, which violates basic assumptions about the prediction error:

- they are unbiased;
- the covariances are equal to the ones computed by the algorithm.

Alternative non-linear filtering algorithms include the UKF [59,60] which uses a set of deterministic sigma points from the Gaussian bell, and sends them through the non-linear model. By doing this, these points will lose their Gaussian distribution, however they will still be approximated by a Gaussian posterior. A further alternative is the Cubature Kalman Filter (CKF) [61], which efficiently computes multivariate moment integrals using the spherical-radial cubature rule.

2.2.2 Sample-Based Approaches

If the true underlying distribution is distinctly non-Gaussian e.g. if the distribution is heavily skewed and/or multimodal, then a *sampling-based approach* may provide a more accurate solution. Rather than having the assumption that the posterior density can be exactly represented by a Gaussian distribution, it may be more appropriate to represent this posterior with a set of weighted discrete samples [62,63]. There are a number of sampling-based approaches and algorithms that fall in to this category, a subset of which will be introduced in the following subsections. The notation used here will closely follow that given in [49].

Grid-Based Methods

The first sample-based approach of interest to this work is the *grid-based method*. A grid-based method can provide an optimal recursion for the filtering process if the *state space is discrete* and the *state space contains a finite number of states* [49]. In situations where we have continuous spaces, we can segment this space into N cells. With this segmenting process, we arrive at an *approximate grid-based method* [63]. Suppose that the prior probability density function (pdf) from $k-1$ is approximated by

$$p_k(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1}) \approx \sum_{i=1}^{N} w_{k-1}^i \delta\left(\mathbf{x}_{k-1} - \mathbf{x}_{k-1}^i\right). \tag{2.18}$$

In order to perform the prediction and update recursion for this method, we use [49, 63]

$$p_k(\mathbf{x}_k|\mathbf{z}_{1:k-1}) \approx \sum_{i=1}^{N} w_{k|k-1}^i \delta\left(\mathbf{x}_k - \mathbf{x}_k^i\right), \tag{2.19}$$

$$p_k(\mathbf{x}_k|\mathbf{z}_{1:k}) \approx \sum_{i=1}^{N} w_k^i \delta\left(\mathbf{x}_k - \mathbf{x}_k^i\right). \tag{2.20}$$

Normally the weights used here would involve solving some integrals, as each cell covers a region of a continuous space. However to simplify the computation further, it can be assumed that the weights are computed at the centre point $\bar{\mathbf{x}}_k^i$ of each cell $\mathbf{x}_k^i$. This reduces the weights calculations to

$$w_{k|k-1}^i \triangleq \sum_{j=1}^{N} w_{k-1}^j p(\bar{\mathbf{x}}_k^i|\bar{\mathbf{x}}_{k-1}^j), \tag{2.21}$$

$$w_k^i \approx \frac{w_{k|k-1}^i p\left(\mathbf{z}_k|\bar{\mathbf{x}}_k^i\right)}{\sum_{j=1}^{N} w_{k|k-1}^j p\left(\mathbf{z}_k|\bar{\mathbf{x}}_k^j\right)}. \tag{2.22}$$

The grid-based method comes with a number of caveats however. The chosen grid must be of a high resolution, or sufficiently dense, so that we can get a good approximation to the continuous state-space. One other disadvantage, which will become apparent in Chapter 5, is that when the dimensionality of this state-space increases, the computational cost of estimating in this space grows exponentially (assuming square cells with equal side-lengths).

Importance Sampling

While the grid-based method provides an optimal solution when the state space is discrete and has a finite number of states, this will not always be the case in practice and alternative sample-based approaches are required. In order to reach the particle filter formulation in the next section, the concepts of *Monte Carlo (MC) integration*

and importance sampling will be introduced first.

Monte Carlo integration provides the basis for Sequential Monte-Carlo (SMC) methods. Suppose that we wish to evaluate a multidimensional integral:

$$Y = \int f(\mathbf{x})\pi(\mathbf{x})\mathrm{d}\mathbf{x}. \tag{2.23}$$

This could potentially be a very difficult integral to solve analytically in practice and hence we may wish to find an MC estimate of this integral. The estimate of the above integral is the sample mean:

$$Y_N = \frac{1}{N}\sum_{i=1}^{N} f(\mathbf{x}^i). \tag{2.24}$$

If the samples $\mathbf{x}^i$ are drawn independently, then Y_N will be an unbiased estimate and will converge to Y according to the law of large numbers.

When considering the Bayesian estimation framework, the density $\pi(\mathbf{x})$ is the posterior density. In practice, it is not often possible to accurately sample from the posterior density itself as it is likely to be non-standard and only known up to a proportionality constant. We can however apply the *importance sampling* method to sample from a *similar distribution* [49]. This similar distribution is often called the *proposal distribution*, denoted as $q(\mathbf{x})$ for the remainder of this section. By correctly weighting the samples drawn from this proposal distribution, the MC estimation process is still viable. As long as $q(\mathbf{x})$ and $\pi(\mathbf{x})$ have the *same support*, the integral in (2.23) can be rewritten as

$$Y = \int f(\mathbf{x})\frac{\pi(\mathbf{x})}{q(\mathbf{x})}q(\mathbf{x})\mathrm{d}\mathbf{x}. \tag{2.25}$$

From this, we can then form an MC estimate of Y by taking samples from $q(\mathbf{x})$ and computing the weighted sum

$$Y_N = \frac{1}{N}\sum_{i=1}^{N} f(\mathbf{x}^i)\tilde{w}(\mathbf{x}^i) \tag{2.26}$$

with the importance weights

$$\tilde{w}(\mathbf{x}^i) = \frac{\pi(\mathbf{x}^i)}{q(\mathbf{x}^i)}. \tag{2.27}$$

This formulation then leads to the particle filter algorithm.

Particle Filtering

By applying the importance sampling method to the non-linear filtering problem, we arrive at the particle filter algorithm [63,64]. The main concept of this filter is to represent the posterior density with a set of weighted random samples, and compute

state estimates based on these samples. As the number of samples increases, this sample-based approximation of the true posterior becomes a more accurate representation of the true posterior, and the particle filter approaches the optimal Bayesian estimator.

Many of the following equations for the particle filter algorithm are closely related to those in the grid-based method and importance sampling above; the full algorithm is presented in [49, pg.38-39]. The samples can follow some chosen dynamical model, and their states can change over time through resampling [65]. Although this technique now lets the particles flow, and almost "search" for the best estimate, there are some new problems that need to be overcome. The first problem is *particle degeneracy*. After a small number of iterations, all of the particles except for one will have very small weights. Because of this problem, a lot of the computational effort is being focused on updating particles that will have little to no effect on the overall estimation of the updated state. It is possible to gain a measure of how much degeneracy is present in the samples by computing an estimate of the *effective sample size* [63, Eq. (51)],

$$\hat{N}_{\text{eff}} = \frac{1}{\sum_{i=1}^{N} \left(w_k^i\right)^2}. \tag{2.28}$$

If the effective sample size falls below a chosen threshold, then a resampling step should be performed to refresh the particles states and weights; however the use of resampling can lead to the further problem of *impoverishment*. Depending on which resampling algorithm is chosen in the design, one could end up replicating the same particle state many times due to its high weighting, and having many particles with low weights. These low-weighted particles will be replaced with the state of the highly-weighted particle, such that all particles could potentially lie on top of one another in one place, leaving little to no variance in the distribution.

2.2.3 Probabilistic Data Association

So far, the STT algorithms presented have assumed an ideal case where the only measurement generated by the sensor corresponds to the single point target in the region. This assumption is now relaxed; the sensor can now generate more than a single measurement at a given iteration. One measurement will belong to the target, with other measurements being considered as false alarms. In order to handle the clutter points, we need a different algorithm; the Probabilistic Data Association (PDA) filter. It incorporates correlation and data association (DA) steps (see Section 2.3.1) to calculate association probabilities between each correlated measurement and the target of interest. In order to keep the algorithm as simple and efficient as possible, after each iteration, a mixture of Gaussians is reduced down to a single Gaussian using moment matching (similar to the GPB approach [57]).

The implementation of PDA is very similar to that of the Kalman filter with

a few extra steps including: a correlation step which will form a list of feasible measurement-to-target associations; the computation of association probabilities for each of these feasible associations; and an inflation of the covariance of the posterior distribution, so that any association uncertainty is accounted for. Depending on the number of feasible associations that are defined, the computational effort required for the PDA method is approximately double that of the Kalman filter approach. This doubling of computational effort holds if only one measurement is inside the gate at each time-step.

The filter first performs the typical prediction step using the chosen dynamical model, followed by the new correlation step. Correlation gates are computed for each measurement-target pair. If the computed gate

$$\nu = (\mathbf{z}_k - h_k(\mathbf{x}_{k|k-1}))' \mathbf{S}_k^{-1} (\mathbf{z}_k - h_k(\mathbf{x}_{k|k-1})), \tag{2.29}$$

where $\mathbf{S}_k$ is the innovation covariance, $\mathbf{z}_k$ is the current measurement and $h_k(\mathbf{x}_{k|k-1})$ is the target state using the appropriate parameterisation, is smaller than some chosen gating threshold, the measurement-target pair becomes a feasible association that we can attempt to resolve next.

The PDA filter uses a probabilistic association scheme to perform its association step; global association/assignment algorithms will be covered in more detail in the next section. For each of the feasible associations, a weight is computed based on a Gaussian likelihood, that the chosen measurement-to-target association is true. We must also take in to account that the measurement is not the correct choice i.e. the measurement is actually a false alarm or clutter point, and does not belong to the target. The association probabilities can be computed by

$$\beta_i = \begin{cases} \frac{\mathcal{L}_i}{1 - p_\mathrm{d} p_\mathrm{g} + \sum_{j=1}^{m} \mathcal{L}_j} & i = 1, \ldots, m \\ \frac{1 - p_\mathrm{d} p_\mathrm{g}}{1 - p_\mathrm{d} p_\mathrm{g} + \sum_{j=1}^{m} \mathcal{L}_j} & i = 0 \end{cases} \tag{2.30}$$

where

$$\mathcal{L}_i \triangleq \frac{1}{p_\mathrm{g}} \frac{\mathcal{N}(\mathbf{z}_i; h_k(\mathbf{x}_{k|k-1}), \mathbf{S}_k) p_\mathrm{d}}{\lambda}, \tag{2.31}$$

$i = 1, \ldots, m$ contain the feasible associations, $i = 0$ represents the no association term, p_d is the probability of target detection, p_g is the gate probability, and λ is the mean number of false alarms per scan. Approximations of these probabilities are important for the MP implementations exploited later in Chapter 5, and in Chapter 6.

Once we have a probability for all of the combinations, a weighted update and state estimation is performed to gain the posterior pdf of the target. These steps are very similar to those found in the Kalman filter update but with a few small changes. Rather than just having the single innovation, or residual term, a combined

innovation y_k that covers all of the measurements must be computed -

$$\mathbf{y} = \sum_{i=1}^{m} \beta_i \mathbf{y}_i \tag{2.32}$$

where

$$\mathbf{y}_i = \mathbf{z}_i - \mathbf{H}\mathbf{x}_{k|k-1} \tag{2.33}$$

The covariance calculation must take account of the multiple measurements that could be associated to the target. The updated covariance is given by

$$\mathbf{P}_k = \beta_0 \mathbf{P}_{k|k-1} + (1 - \beta_0)\mathbf{P}_k^C + \tilde{\mathbf{P}}_k, \tag{2.34}$$

where the covariance of the update with the correct measurement is

$$\mathbf{P}_k^C = \mathbf{P}_{k|k-1} - \mathbf{K}\mathbf{S}\mathbf{K}' \tag{2.35}$$

and $\mathbf{K}$ is the Kalman gain. The covariance of the posterior pdf is also increased by the spread of the innovations term (due to association uncertainty) [21]

$$\tilde{\mathbf{P}}_k \triangleq \mathbf{K}\left[\sum_{i=1}^{m} \beta_i \mathbf{y}_i \mathbf{y}_i' - \mathbf{y}\mathbf{y}'\right]\mathbf{K}'. \tag{2.36}$$

This filter provides a platform for moving on to MTT scenarios next, as it contains the key elements of an MTT algorithm. The PDA filter can be extended to the Joint Probabilistic Data Association (JPDA) filter; a common vector-type method found in practice.

2.3 Multi-Target Tracking and Fusion

The MTT problem is a very important problem in practice, and is now prominent in many applications. Much of the early development work in the area was heavily driven by aerospace applications, where the aim was to track hundreds of targets from radar measurements. At each time step k, a set of noisy measurements is generated by the sensor and passed in to the tracking algorithm. It is up to the tracking algorithm to make sense of these measurements as there are likely to be false alarms and missed detections.

In the general case, the MTT problem now includes a model for target birth and target death. The process of initiating and terminating tracks is orthogonal to the multi-target tracking problem. In single target tracking, we know that there will either be zero, or at most one, target. In MTT, we know that there are, with high probability, $N > 1$ targets. The birth model is required so that new targets entering from the edges of the state space, or targets that have not previously been detected

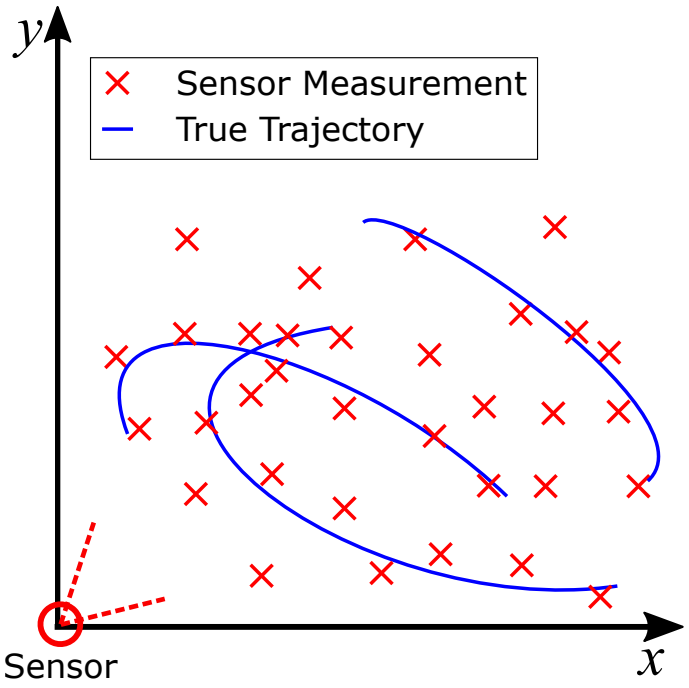


Figure 2.1: An example of an MTT scenario.

in the state space (e.g. hidden/stealthy/obscured targets) can be tracked. We also require a death model so that targets leaving the state space can be appropriately modelled. These form a part of the multi-target dynamical model. Each target $\mathbf{x}_{k-1}$ that exists from the previous time-step $k-1$ will either continue to exist at time-step k, with probability p_{s}, and move to a new state according to the chosen transition model $f_{k|k-1}(\mathbf{x}_k|\mathbf{x}_{k-1})$, or the alternative is that the target will no longer exist at time k with probability $1-p_{\mathrm{s}}$. Each target is assumed to evolve and appear/disappear independently of the other targets. Other variables could also be included in the multi-target model and propagated over time, such as the probability of target existence [66]; a variable commonly found in Bernoulli-type processes and filters. This probability could act as a basis for the birth/death model in practice; if this probability is higher/lower than a given threshold, we could say that a target either does, or does not, exist.

The last element is the modelling of new targets. There are a number of different methods for initialising new target states, such as measurement-driven birth [67], where we could place new states on all measurements; we could exploit correlation and association information. If a measurement does not correlate or associate with any current targets, this may be the first time we have observed it so we will place a new state on that location. Further alternatives could take spatial or contextual information into account such as randomly placing new states across the space to cover the full region; locate them close to a region where we would expect more targets to be present, e.g. ships entering/exiting a harbour, on roads etc; or by initialising a large distribution across the whole space with a given birth weight.

For the standard multi-target observation model, each target $\mathbf{x}_k$ surviving to

time k will either be detected with probability p_{d} and generate a sensor measurement $\mathbf{z}_{k,m} \in \mathbf{Z}_k$ with a likelihood $l_k(\mathbf{x}_k|\mathbf{z}_{k,m})$, or not detected with probability $1 - p_{\mathrm{d}}$. It should be acknowledged that the probability of detection p_{d} is potentially time-varying and state-dependent; however for simplicity and brevity, we will assume that it is constant throughout. As well as the target measurements, $\mathbf{Z}_k$ will likely contain a number of false alarm measurements. False alarm rates tend to follow the Poisson distribution, with a uniform spatial distribution across the measurement space [19, 21]. Some alternative clutter models will be explored in Section 2.3.3 where the Panjer filter is introduced. The standard observation model assumes that each measurement can only be generated from one target, i.e. a measurement cannot be associated to more than one target, and each target will generate a measurement independently from other targets [19]. Although this is the standard model, other models do exist in practice which can take other situations in to account such as extended targets [68] and unresolved targets [69].

2.3.1 Association Algorithms

Before some common MTT algorithms are introduced, there is one further aspect that needs to be examined in more detail; the *association problem*. This was briefly introduced in Section 2.2.3 for the single-target probabilistic case, however this needs to be extended for the multi-target case, where we need to resolve the global assignment problem.

Again, a correlation step is used to remove all of the measurement-to-target pairs whose gate is larger than the chosen gating threshold (see Section 2.2.3), which will save on computation time during the association step. The correlation window size and shape can be designed depending on the application. For example, this window could be a circular or elliptical shape, based on a volume defined by the chi-square distribution with n_z degrees of freedom for a general surveillance scenario. Another example could be to use a rectangular correlation window, which is often used in situations where the state space and the measurements are both in Cartesian coordinates.

In an ideal situation, only one measurement would lie inside the correlation region for each individual target (Figure 2.2); this would make the association problem very straightforward as there is only one feasible measurement-to-track pair in the immediate region. Of course it should be noted that this feasible measurement, as defined through the correlation process, could be an incorrect one and not match the true measurement in practice. However, this rarely happens in practice and we often see multiple measurements inside the region for each target (Figure 2.3). This could be due to false alarms, or a target that has generated multiple measurements (the target may be significantly larger than the range and azimuth resolution of the radar [36]). The case of multiple measurements per target would break the underlying

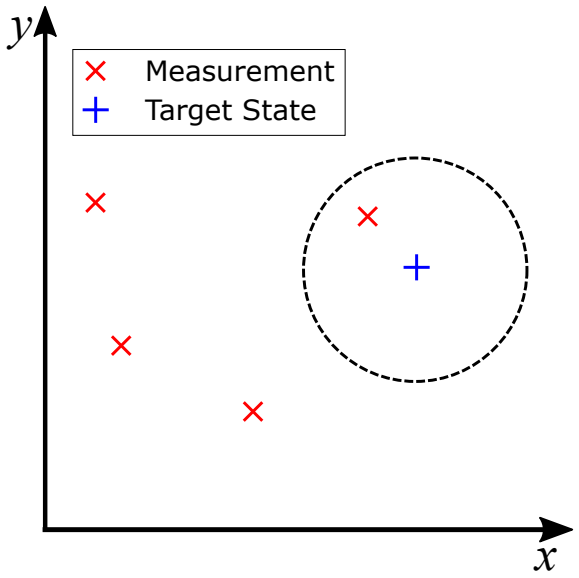


Figure 2.2: Simple correlation-association example, only one feasible candidate for association to the target. The ellipse represents the correlation gate.

assumption of point targets, i.e. at most one measurement being generated by a single target. In practice, a clustering, collapsing, or centroiding method could be used to reduce these down to a single measurement, or develop an appropriate observation model such that extended targets can be tracked [68]. It should be noted that these clustering routines are not MTT-specific and could also be applied in the STT environment if the problem space required them. For the work presented in this thesis, it will be assumed that all targets are point targets, and each of these will only generate a single measurement. We need to make a decision on which of these feasible associations should be assigned for the update step.

There are a number of algorithms available for resolving this DA (or assignment) problem when we have a number of feasible measurement-target associations. The simplest approach is to use the *nearest neighbour approach*, which will assign the measurement that is closest to a predicted state. Any appropriate distance metric could be used here depending on the state and measurement representation, e.g. Euclidean, Mahalanobis, etc.). Although this is simple and easy to implement, it is trivial to see the issues that arise in practice with this approach. The optimal solution may often not be found, and this method is very poor when we consider dense tracking scenarios with many closely-spaced targets. This nearest neighbour approach can be extended to the Global Nearest Neighbour approach, which is discussed in more detail in the next section.

A very popular association algorithm is the Hungarian algorithm [70], often more commonly known as the Munkres algorithm [71]. It was developed in the mid-1950s and remains a common and popular choice when developing target tracking methods. By performing a number of repeated row and column operations to the cost matrix (a matrix populated with (weighted) distances between measurement-target

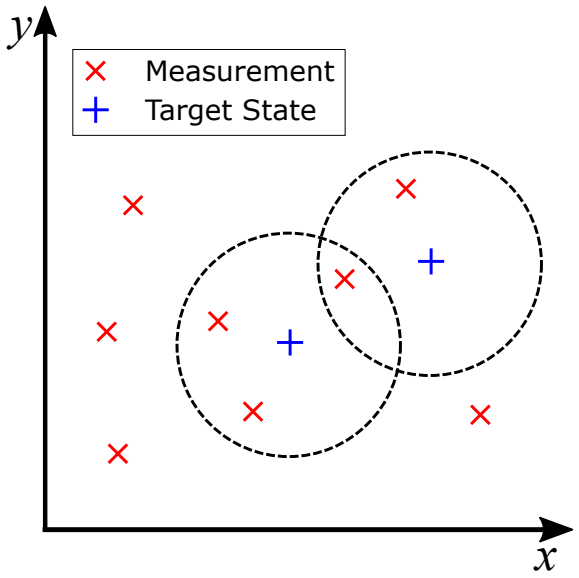


Figure 2.3: Difficult correlation-association example, one target has three candidates, and the other targets has two candidates. One measurement is in the overlapping region, but can only be associated to one of the targets.

pairs), the optimal assignment of measurements to targets can be found. There are a number of limitations with this method however; it assumes that the matrix dimensions are equal (a balanced, or square, matrix) i.e. the number of measurements to be associated is equal to the number of targets, giving a 1-1 mapping. Without some intensive pre-processing of measurements and strict correlation thresholds, this situation can be difficult to reach, especially when considering the dense scenarios. One alternative for resolving this issue in practice may be to populate the smaller dimension with a set of dummy points that are sufficiently far away from the true points. This should not affect the association result, and will balance the dimensions of the matrix. A later extension to the Munkres algorithm in the early-1970s [72] adapted the algorithm to allow for assignment in rectangular matrices.

An alternative to Munkres is the Auction algorithm [73], which builds on a number of the features of Munkres, but is much more efficient, and converges to the optimal assignment at a much faster rate. Auction is aptly named, as the steps leading to the optimal assignment resemble a typical bidding process. Each of the measurements effectively "bids" repeatedly for targets to be associated to. After the "price" for a track reaches a level where no further bidding by any of the measurements would be profitable, the algorithm will stop, and the "winners" receive their tracks. The original Auction algorithm was also only designed for assignment in square matrices much like the original Munkres implementation. An alternative called the Forward/Reverse Auction algorithm [74] was developed to extend Auction to deal with rectangular matrices. A comparison of these techniques, along with some brute force methods (where every possible measurement-to-track pair is assessed for suitability before a decision is made) is given in [75]. Their full

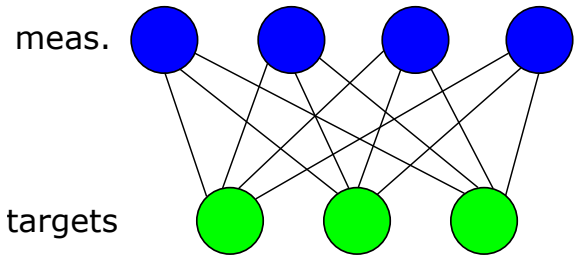


Figure 2.4: A complete bipartite graph, all associations are feasible.

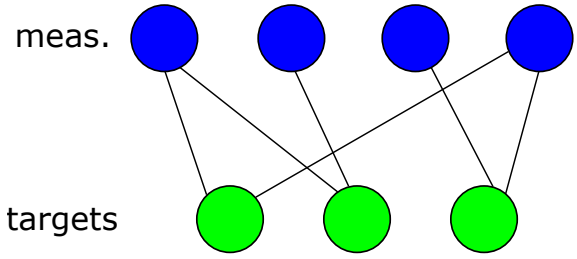


Figure 2.5: Bipartite graph, post-correlation process. Some associations are no longer feasible.

mathematical framework and derivations are provided in [76].

A fourth method, which will be exploited as a part of this work later in Chapter 5 and Chapter 6, is the Sum-Product Algorithm for Data Association (SPADA) [77]. It exploits the power and efficiency of Bayesian graphical models to resolve the DA problem very quickly. The association problem is created on a bipartite graph structure (see Figure 2.4 and Figure 2.5).

A bipartite graph is a graph with vertices that can be split into two disjoint and independent sets, so that every edge of the graph connects one vertex in set A to one vertex in set B. This fits the association problem perfectly, with one set containing measurements and the other set containing tracks. It should be noted that both the Munkres and Auction algorithms could also be formulated using bipartite graphs. Belief Propagation (BP) is used to approximate the association probabilities between measurements and targets, and the Sum-Product Algorithm (SPA) passes messages around a bipartite graph structure that represents the problem. After a set number of Message Passing (MP) iterations have been completed, or a convergence criterion has been reached, the algorithm stops and beliefs are computed for each associated pair. This method has been shown to be very efficient in comparison to Markov Chain Monte-Carlo (MCMC) methods for data association [23]. An alternative to the SPA is the Max-Product Algorithm. Rather than attempting to find and solve the marginals, the Max-Product algorithm attempts to maximise a global function, and works with Maximum A Posteriori (MAP) estimates. Working with MAP estimates can be more erratic, as the sample with the highest weighting will likely change quite rapidly over time. When using the weighted mean, the state estimate will likely be much smoother over time and be much more consistent. Because of

this, the SPA is often preferred. The algorithms themselves are almost identical [78].

One further interesting idea may be to try and associate over windows of time, rather than at individual time-steps using efficient schemes such as the one presented in [79].

2.3.2 Vector-Type Methods

Now that all of the appropriate models and functions have been introduced, some popular and state-of-the-art methods for performing MTT will be briefly summarised. MTT algorithms are broadly split in to two categories; vector-type methods and point process methods. Vector-type methods will be addressed first.

Vector-type methods allow for each *individual target state to be distinguishable* from the overall population, i.e. it has a unique target identity, and can be treated as an individual [80]. The simplest algorithm that falls into this category is the Global Nearest Neighbour (GNN) filter, which is an extension of the nearest neighbour algorithm discussed earlier [21]. This filter will search for the optimal solution when assigning measurements to targets by attempting to either maximise or minimise a total cost function, such as weighted distances or likelihoods. Once the filter has performed this step, it then uses the standard Bayes update on the associated measurement for each track. Again, this is a simple scheme that would only require a simple filter with an association algorithm, however it has the same pitfalls as the nearest neighbour algorithm [81]:

- poor performance when we have a dense scenario with many closely spaced targets;
- computationally expensive to run;
- susceptible to track loss and track splits.

The next vector-type algorithm is an extension of the PDA filter presented in Section 2.2.3, namely the Joint Probabilistic Data Association (JPDA) filter [21,82,83]. The filter is designed to deal with a fixed and known number of targets, which can potentially limit its use in practice. When considering the multiple target case, JPDA uses joint association events and joint association probabilities to avoid issues with conflicting assignments during the track update step. Determining these events and probabilities can become a very cumbersome task especially with a larger number of targets. There have been a number of attempts to try and reduce this computational expense by using approximations [2, 84–86] and alternative association algorithms such as SPADA [77]. As the original JPDA algorithm was only developed for a fixed and known number of targets, a further extension to the Joint Integrated Probabilistic Data Association (JIPDA) filter [87] allows for an unknown and time-varying number of targets, as well as models for target birth and target

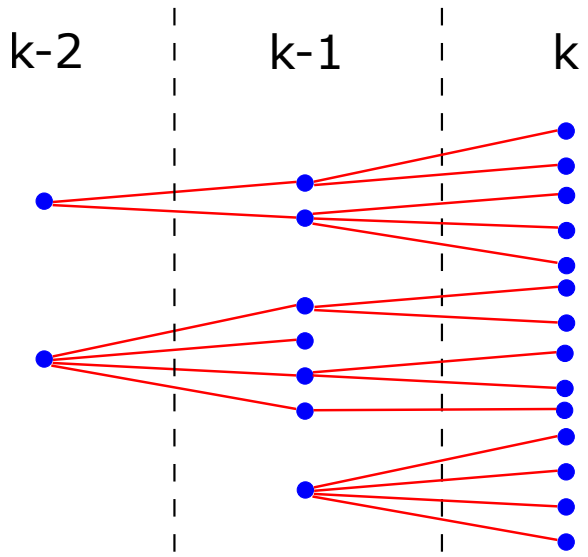


Figure 2.6: Example of an Multiple Hypothesis Tracking (MHT) structure.

death. The JPDA and JIPDA filters are of interest, as the MP algorithm introduced in Chapter 5 is a MP-based approximation of these filters [22, 23]. An introduction to MP and factor graphs is provided in the next section.

The last vector-type algorithm is that of Multiple Hypothesis Tracking (MHT) [88–90], a deferred decision approach to MTT, and one of the oldest and most robust techniques that could be used. The MHT structure grows over time in to a form of tree diagram as shown in Figure 2.6. At each time-step k, the algorithm considers the association of sequences of measurements and evaluates the probability of all of the potential association hypotheses. When the new set of measurements is fed into the algorithm, a new set of hypotheses is generated from the hypotheses at time $k-1$, and the probability is then computed using Bayes rule. Once we have selected our most probable tracks (this can be very computationally expensive as it involves an exhaustive search), a Kalman filter update can be used on the individual track to give the new trajectory. The use of hypothesis management is critical when attempting to implement a form of MHT scheme in practice as in general, MHT scales exponentially over time. This could involve the pruning or removal of hypotheses that have a low probability, the merging of hypotheses that have current states which are located very close to one another, or only considering a small window of time, rather than maintaining hypotheses from time 1 to time k.

2.3.3 Point Process Methods

The alternative to vector-type methods are the more recently developed point process methods, which have gained a lot of interest in the target tracking literature. They have gained more interest as they have shown to be more efficient in a number of applications [23], and the potentially costly association problem is avoided [91].In this case, although the target states and measurements are still treated as random

vectors, the point process methods do not *distinguish or identify individual targets* in the larger population [80]. This is a useful tool when analysing observed patterns of points, where the points could represent the locations of some objects, e.g. measurements on a radar screen. Point processes can be described by some chosen discrete distribution. For example let us consider a Poisson point process; the cardinality is Poisson distributed with a given mean, and the points will be independent and identically distributed (i.i.d.) according to some chosen distribution (uniform, Gaussian, ...). This approach to MTT still retains the Bayesian methodology seen with vector-type methods.

Point Processes and Probability Generating Functionals (PGFLs)

The Probability Hypothesis Density (PHD) filter, its generalisations, and the Multi-Object Likelihood (MOL) functions introduced later in this section, can be derived from *point process theory.* The Finite Set Statistics (FISST) theory proposed by Mahler in his seminal works [92,93], along with an Random Finite Set (RFS) representation, provides an alternative formulation to the point process approach applied in this thesis. A *point process* denoted Φ on $\mathcal{X}$ is a random variable on the space $\mathfrak{X} = \bigcup_{n\geq 0} \mathcal{X}^n$ of finite sequences in $\mathcal{X}$. A single realisation of Φ is a sequence $\varphi = (x_1, \ldots, x_n) \in \mathcal{X}^n$, where the number $n \in \mathbb{N}$ and the states $x_i \in \mathcal{X}$ of the objects are random.

The main point process that is required, in order to reach the PGFLs that are presented in Appendix A, is the i.i.d. cluster process:

Definition 2.3.1 (Independent and identically distributed cluster process)**.** *An independent and identically distributed (i.i.d.) cluster process with cardinality c and spatial distribution $s_\star$ describes a group of objects with size described by c and i.i.d. states according to $s_\star$. Its PGFL is given by*

$$G_{iid}(h) = \sum_{n\geq 0} c(n) \left[\int h(x) s_\star(x) \mathrm{d}x \right]^n, \tag{2.37}$$

where h is a real-valued function.

The i.i.d. cluster process is general as it does not make an assumption about the cardinality distribution c [94,95]. By substituting in the appropriate cardinality distribution, it is possible to gain the PGFLs defined in Appendix A. The first cardinality distribution of interest to this work is the Bernoulli distribution:

$$c(0) = 1 - p, \quad c(1) = p, \quad c(n) = 0 \text{ for } n > 1. \tag{2.38}$$

A Bernoulli process can be used to describe binary events such as the detection and survival of individual targets in the surveillance region, as they only have two

possible outcomes; either, the object no longer exists (or is not detected by the sensor) with probability $(1-p)$, or the object continues to exist (or is detected by the sensor) with probability p. The second important distribution is the Poisson distribution

$$c(n) = \frac{\lambda^n e^{-n}}{n!}, \tag{2.39}$$

which has found popularity in the community, as it can describe common phenomena in a range of applications, while only being dependent on one parameter. The exponential form also makes it easy to work with. When constructing the PHD filter, Poisson processes are used to represent the predicted target and clutter processes. The final distribution of interest is the Panjer distribution [95, 96]

$$c(n) = \binom{-\alpha}{n} \left(1 + \frac{1}{\beta}\right)^{-\alpha} \left(\frac{-1}{\beta+1}\right)^n. \tag{2.40}$$

The Panjer process has not often been used in the engineering community previously, but has been widely used in the actuarial mathematics area [96]. The Panjer process allows much more flexibility in the underlying model choices, particularly in the case of false alarm distributions as explored later in this section. More information on the Panjer distribution and how it can be implemented in the MTT environment can be found in [95, 97].

By using the PGFLs and mathematical tools given in Appendix A, it is possible to derive the the PHD filter, and its generalisations such as the Panjer filter.

The PHD filter

The original filter developed in the RFS literature is the Probability Hypothesis Density (PHD) filter [92], with the commonly-found Gaussian mixture implementation coming later in 2006 [91]. The PHD filter is a first-moment approximation which alleviates the computational intractability of the multi-target Bayes filter. By propagating just the first moment (mean number of targets), the PHD filter operates on a single-object state space, and avoids dealing with the data association problem. The full recursion is defined by two equations, which still follow the Chapman-Kolmogorov equation and Bayes rule:

$$\mu_{k|k-1}(\mathbf{x}_k) = \mu_{b,k}(\mathbf{x}_k) + \int p_{\mathrm{s}}(\mathbf{x}_k) f_{k|k-1}(\mathbf{x}_k|\mathbf{x}_{k-1}) \mu_{k-1}(\mathbf{x}_{k-1}) \mathrm{d}\mathbf{x}_{k-1} \tag{2.41}$$

$$\mu_k(\mathbf{x}_k) = \left[1 - p_{\mathrm{d}}(\mathbf{x}_k)\right] \mu_{k|k-1}(\mathbf{x}_k) + \sum_{\mathbf{z}\in\mathbf{Z}_k} \frac{p_{\mathrm{d}}(\mathbf{x}_k) l(\mathbf{x}_k|\mathbf{z}) \mu_{k|k-1}(\mathbf{x}_k)}{\mu_{c,k}(\mathbf{z}) + \int p_{\mathrm{d}}(\mathbf{x}_k) l(\mathbf{x}_k|\mathbf{z}) \mu_{k|k-1}(\mathbf{x}_k)} \tag{2.42}$$

where $\mu_{k|k-1}(\mathbf{x}_k)$ is the predicted PHD intensity, $\mu_k(\mathbf{x}_k)$ is the updated PHD intensity, $\mu_{b,k}(\mathbf{x}_k)$ is the intensity of the birth point process (a Poisson point process and independent of the surviving objects point processes) and $\mu_{c,k}(\mathbf{z})$ is the intensity of

the clutter point process (a Poisson point process and independent of the object generated measurement point processes). The predicted and updated multi-object point processes are also approximated by Poisson point processes. By avoiding resolving the data association problem, the PHD filter is a very fast algorithm, but this comes at a cost of losing a history of states and the inability to draw target tracks, unless extra elements such as labelled sets or a Gaussian tagging scheme are introduced (see explanation of Generalised Labelled Multi-Bernoulli (GLMB) filter below). This relates back to the idea of *distinguishability* of targets that was introduced earlier. The PHD update step also includes mixture reduction techniques similar to those found in the MHT algorithm described previously. These include a pruning step where Gaussians that have low weights are deleted from the mixture, and a merging step where Gaussians are merged together if they are located close to one another, or significantly overlap, depending on which type of distance measure is used e.g. Mahalanobis, Hellinger [98] etc. One further drawback is the assumption that the pdf is formed from the cluster process approximation. Because of more assumptions, performance and accuracy are likely to be reduced.

A Sequential Monte-Carlo (SMC) implementation of the PHD filter was first developed in 2005 [99]. This particle-based implementation of the PHD filter can work directly with non-linearities, such as the Cartesian to Polar transformation between spaces. Directly applying typical SMC methods to propagate the PHD intensity would fail as it is not strictly a pdf. Instead, the intensity function is represented by a large set of weighted particles which are propagated over time using a generalised importance sampling and resampling strategy. The number of particles in this set can be continually adapted, depending on the estimated number of targets in the surveillance region [99]. For example, we could inject more particles into the space if we think more targets have appeared in the region, or if the uncertainty in the existing target states have grown.

A further implementation that could be applicable to such problems was also first presented in 2005 [100]. While mainly developed for tracking extended targets which generate multiple measurements, the models that this implementation is based on are very suitable for particle-based implementation. It can also be extended to the multiple target tracking problems if required.

PHD Filter Generalisations/Extensions

A further generalisation of the PHD filter is the Cardinalized PHD (CPHD) filter [94, 101]. Rather than propagating just the moments of the cardinality distribution, the CPHD filter propagates both the PHD and the full cardinality distribution, which gives higher tracking accuracy but requires much higher computational expense. However, a more recent addition to the literature provides a form of "middle ground" between the PHD filter and the CPHD filter. The Panjer filter was first

introduced in [97] as a useful extension to the first-order PHD filter defined earlier. Rather than only propagating the first-order moment, the Panjer filter can propagate both the mean and variance of the point process, under the assumption that the number of predicted targets and the false alarms are Panjer distributed. The Panjer distribution is characterised by two parameters α and β which closely correspond to its mean and variance. Depending on how these parameters are set, three different distributions can be implied:

Case 1: $\{\alpha, \beta\} \in \mathbb{R}^+ \times \mathbb{R}^+$**,** represents a negative binomial distribution, whose variance is greater than the mean. With the variance much larger than the mean, we are likely to see much more variation in the number of sensor measurements being generated at a given iteration. Having this false alarm distribution available would be useful in situations where there may be sudden large influxes of measurements such as strong returns on a rough sea [48, 102]; or in built-up urban environments where there may be multiple measurements generated due to multipath effects from buildings and objects [36].

Case 2: $\{\alpha, \beta\} \in \mathbb{Z}^- \times \mathbb{R}^-$**,** represents a binomial distribution where the variance is less than the mean. With the smaller variance, we are likely to see a much more consistent false alarm rate, and in practice, maintain a constant probability of false alarm p_{FA}. This type of false alarm distribution could be used in application areas that have near-static or slow-changing target dynamics. Another potential use case for this distribution may be in ground-based radar where the sky is being scanned for targets. Relating back to case 1 above, if an airborne radar is attempting to detect targets on the ground, it is likely to get returns from many bright objects such as buildings etc. This is not the case for ground-based radar, as there are very limited objects to detect in the sky, and therefore a much more consistent number of false alarms generated. It should be noted that this is dependent on the frequency that the radar operates at.

Case 3: $\{\alpha, \beta\} \to \infty$**,** $\frac{\alpha}{\beta} = \lambda = const.$**,** is the limit case where the ratio stays constant, resulting in the Poisson distribution. This represents the standard false alarm model used in many pieces of target tracking literature.

Having these three different cases available gives more flexibility in modelling the number of targets and false alarms. A full mathematical derivation and pseudo-code is available elsewhere [97, 103].

There has been a lot of recent attention in labelled methods such as the GLMB filter [104, 105]. By including labels with the target states, it is then possible to maintain a unique identity for each target, and therefore gain distinct track histories. Each target is labelled with an ordered pair of integers; firstly by the time that the target was born and secondly a unique index so that targets born at the same time-step can be distinguished from one another. This removes any ambiguity over

numbering conventions and this keeps a consistent label over time that does not require changing or updating. These labelled methods almost extend the functionality that we see within the PHD filter, which in its standard implementation has no ability to keep a track of target identity. Filters such as the GLMB are however based on a lot of the RFS and FISST foundations [92] that also form the PHD filter. By including these labels, the high-level features offered by the labelled-RFS methods are similar to those offered by vector-type methods such as MHT for example.

These labelled methods, however, come with a significantly higher computational cost than the standard PHD filter, making them less suited for practical implementation on a mobile system where we need scalable and efficient algorithms.

Single-Cluster Methods

The solution that is proposed for resolving the sensor registration and tracking problems is a *single-cluster method.* Cluster processes describe a hierarchical framework of *point processes* where one realisation of an offspring process is conditioned on the realisation of some parent process [106]. In other areas of the literature, this may be referred to as a *doubly-stochastic process* [107], or a *single-cluster point process* [106, 108]. An alternative to the single-cluster method is the Rao-Blackwellised (RB)-PHD filter, which was introduced for the Simultaneous Localisation and Mapping (SLAM) application in [109]. Rather than the assumption of hierarchical point processes that is used in the single-cluster method, the RB-PHD filter requires Poisson approximations on both the prior, the posterior, and for the SLAM application, an approximation on the number of features. Because of the extra approximations made in the RB-PHD filter, the single-cluster method should give a more realistic representation of the true multi-object distribution [106].

Single-cluster methods are a very flexible framework, which could in theory, allow for any form of MTT algorithm to be included in the offspring process, as along as an appropriate MOL function was derived to link the processes together. The single-cluster method is closely related to Bayesian hierarchical models, which are more commonly found in the vector-type literature. This single-cluster method will be compared to a Bayesian hierarchical model implementation later in Chapter 5.

A single-cluster process can be represented using a state variable given by $\mathbb{X} = (\mathbf{q}, \mathbf{X})$, where $\mathbf{q}$ is the state vector for the cluster centre in the parent process state-space, and $\mathbf{X}$ contains the set of target state vectors that are being tracked in the offspring process.

Let $\mathbf{Z}_{1:k}$ be the sets of sensor measurements available at time-steps $1, \ldots, k$. Each of the measurement sets contains a random number of measurement vectors which will have been generated by some pre-processing inside each sensor. Let us suppose that the prior density $p_{k-1}(\mathbb{X}_{k-1}|\mathbf{Z}_{1:k-1})$ is known, and a new set of sensor measurements are received. A Bayes recursion can be used to propagate the posterior

density, i.e.

$$p_{k|k-1}(\mathbb{X}_k|\mathbf{Z}_{1:k-1}) = \int f_{k|k-1}(\mathbb{X}_k|\mathbb{X}_{k-1})p_{k-1}(\mathbb{X}_{k-1}|\mathbf{Z}_{1:k-1})\mathrm{d}\mathbb{X}_{k-1} \tag{2.43}$$

$$p_k(\mathbb{X}_k|\mathbf{Z}_{1:k}) = \frac{\hat{\ell}_k(\mathbb{X}_k|\mathbf{Z}_k)p_{k|k-1}(\mathbb{X}_k|\mathbf{Z}_{1:k-1})}{\int \hat{\ell}_k(\mathbb{X}'|\mathbf{Z}_k)p_{k|k-1}(\mathbb{X}'|\mathbf{Z}_{1:k-1})\mathrm{d}\mathbb{X}'} \tag{2.44}$$

where $f_{k|k-1}(\mathbb{X}_k|\mathbb{X}_{k-1})$ is a Markov transition function for the states in both the parent and offspring processes, and $\hat{\ell}_k(\mathbb{X}_k|\mathbf{Z}_k)$ is a MOL function [110, 111]. The MOL functions that will be exploited in Chapter 4 are presented next. The above equations follow the Chapman-Kolmogorov equation and Bayes' rule and represent the general single-cluster methodology. These equations can however be broken down into two interleaved recursions for the parent and offspring processes as shown in Chapter 3, and in [35].

Multi-Object Likelihoods (MOLs)

The MOL is an important likelihood function which will be applied in both Chapter 4 and Chapter 5. These likelihoods will provide the fundamental link between the sensor registration problem and the sensor fusion problem. An appropriate MOL has been derived previously in the literature for both the PHD filter and the Panjer filter; both of which will be exploited in Chapter 4. Proofs of these MOL functions can be found in the supplementary document of [35].

PHD Filter MOL

For the original PHD filter, the MOL function was first derived in [110, 111], and for a given sensor registration $\mathbf{q}$ is

$$\hat{\ell}(\mathbf{q}_k|\mathbf{Z}_k) = \frac{\prod_{\mathbf{z}\in\mathbf{Z}}\left[\mu_c(\mathbf{z}) + \int_{\mathcal{X}} p_\mathrm{d}(\mathbf{x})l(\mathbf{x}|\mathbf{z},\mathbf{q})\mu_{pr}(\mathbf{x})\mathrm{d}\mathbf{x}\right]}{\exp\left[\int_{\mathcal{Z}}\mu_c(\mathbf{z})\mathrm{d}\mathbf{z} + \int_{\mathcal{X}} p_\mathrm{d}(\mathbf{x})l(\mathbf{x}|\mathbf{z},\mathbf{q})\mu_{pr}(\mathbf{x})\mathrm{d}\mathbf{x}\right]}, \tag{2.45}$$

where $l(\mathbf{x}|\mathbf{z},\mathbf{q})$ is the single-object likelihood, $\mu_c(\mathbf{z})$ is the clutter intensity and $\mu_{pr}(\mathbf{x})$ is the predicted intensity. The implementation of this MOL is taken from the work presented in [95, 112]. The full derivation of this can be found in the appendix of [111].

Panjer PHD Filter MOL

First, we should define the Pochhammer symbol or rising factorial $x_{n\uparrow}$ [97, 113] by

$$x_{n\uparrow} := \prod_{i=0}^{n-1}(x+i), \quad x_{0\uparrow} := 1. \tag{2.46}$$

and define α_c, β_c as the Panjer clutter parameters and s_c as the spatial distribution for the clutter. We can then write $\alpha = \alpha_{pr}, \beta = \beta_{pr}$ and s_{pr} as the Panjer parameters and spatial distribution of the predicted process, and also let

$$F_{d,q} = 1 - \frac{1}{\beta} \int_{\mathcal{X}} p_{\mathrm{d}}(\mathbf{x}|\mathbf{q}) s_{pr}(\mathbf{x}|\mathbf{q}) \mathrm{d}\mathbf{x}, \tag{2.47}$$

and

$$F_c = 1 + \frac{1}{\beta_c} \tag{2.48}$$

be two auxiliary functions for a given registration configuration q. The MOL function for the Panjer filter [95, 112] can then be shown to be

$$\hat{\ell}(\mathbf{q}_k|\mathbf{Z}_k) = \sum_{j=0}^{|\mathbf{Z}|} \frac{\alpha_{j\uparrow}}{\beta^j} \frac{(\alpha_c)_{(|\mathbf{Z}|-j)\uparrow}}{(\beta_c+1)^{|\mathbf{Z}|-j}} F_{d,q}^{-\alpha-j} F_c^{-\alpha_c-|\mathbf{Z}|-j} \sum_{\substack{\mathbf{Z}' \subseteq \mathbf{Z} \\ |\mathbf{Z}'|=j}} \prod_{\mathbf{z} \in \mathbf{Z}'} \mu_{\mathbf{z}}(\mathcal{X}|\mathbf{q}) \prod_{\mathbf{z}' \in \mathbf{Z} \setminus \mathbf{Z}'} s_c(\mathbf{z}|\mathbf{q}). \tag{2.49}$$

2.3.4 Radar and Image Fusion

There have been a number of articles in the literature that have investigated sensor fusion for both radar and imaging sensors, across a range of different application areas. Firstly in 2009, Wu et al. [26] proposed an algorithm for performing distributed fusion between a 24 GHz radar and a stereo camera system for an autonomous driving and driver assistance application. The scenario presented contains a single vehicle with the sensors attached and a single target vehicle. The algorithm presented in this article performs a crude nearest neighbour association step by finding the smallest Mahalanobis distance between the closest point detected in the camera system, and the radar measurements.

Moving ahead to 2011, the algorithm presented in [28] provides a means of performing fusion between radar and infrared sensor measurements. It is based on a sequential Gaussian Mixture (GM)-PHD filter, often referred to as the iterator-correcter PHD filter in the wider MTT literature [114]. The scenario contains two dynamic targets which follow a CT model, and exploits the more accurate angular measurements coming from the infrared system to give a more accurate tracking output.

In an alternative application area of avian monitoring, work has been carried out to perform fusion between radar, infrared and acoustic based sensors [27]. The fusion architecture presented in this article operates in two different stages; firstly the radar and infrared data is fused at the feature level, which is then fused with the acoustic data at the higher decision level. Correlation and association steps are carried out during the feature level fusion to find the most probable pairs of features in the radar and infrared data. In order to make the decisions at the second level of

fusion, a fuzzy Bayesian system and Bayesian inference are exploited. The acoustic data is used to aid target identification and classification, and is then fused with the appropriate target track gained in the feature level fusion.

The application of multistatic radar [43, 44] is of great interest for defence and surveillance applications. The use of radar allows for large volumes of space to be scanned in short periods of time. The work carried out by Yan et al. [25] proposes a scheme for defining whether or not the use of distributed fusion in multistatic radar is justified, or if the single sensor tracking estimates are satisfactory. This is carried out using the Cramer-Rao lower bound, a fusion rule-set, and parameter choices set by the user. The performance benefits of fusion may be limited in certain cases by higher measurement uncertainty for example; so it may not always be beneficial to fuse the current measurements. The set of rules presented in [25] could be a useful addition to practical systems that exploit distributed fusion architectures.

A very recent addition to the radar and infrared fusion literature is the work by Mallick et al. in [24]. A key part of track fusion is having the ability to compute the cross covariance; this may not always be possible due to the local trackers not operating with the same dynamic models, or having the same state vector representation. Because of the different state representations, or because a different dynamic model is operating in each individual sensor, the fusion problem becomes much more difficult. For example, the radar may be using a typical NCV model, and the infrared system could contain a constant turn model; these would predict the target in a very different location. The units of the process noise in the active tracker found in the radar, and the passive tracker found in the Infrared Search and Track (IRST) sensor will also not be consistent and cannot directly be fused. The authors propose a Cartesian state vector in the radar tracker containing three-dimensional position and velocity components, and a modified-spherical coordinates state vector for the IRST tracker. By performing local tracking in each of the sensors using a CKF, it is then shown to be possible to perform track fusion using the information filter, which can appropriately handle the issues arising from the process noise inconsistency. The main features, and omissions, from each article are tabulated in Table 2.1. It can be seen that although the radar and image fusion literature presents a range of different real-world applications; none of these articles address or even mention the sensor calibration or registration problem. As eluded to earlier, this almost paints the picture that the sensor fusion process is trivial and bias-free in practice which is definitely not the case.

2.4 Factor Graphs

As a precursor to the work presented in Chapter 5 and Chapter 6, this section provides an overview of factor graphs, and Message Passing (MP) algorithms. The

Table 2.1: Radar and Image Fusion Summary

Article	Measurement Fusion	Multi-Target	Heterogeneous Sensors	Sensor Registration
Wu et al. [26]	✗	✗	✓	✗
Huazhi and Jian [28]	✓	✓	✓	✗
Mirzaei et al. [27]	✓	✗	✓	✗
Yan et al. [25]	✗	✗	✗	✗
Mallick et al. [24]	✗	✗	✓	✗

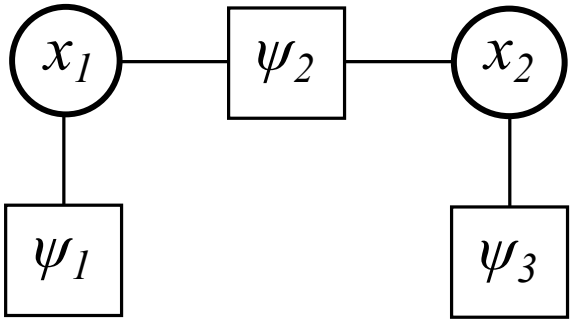


Figure 2.7: A simple factor graph example.

statistical structure of a problem can be described using a *factor graph*. It is often the case that algorithms will deal with a large and complex global function that may be difficult to solve, and be dependent on many variables. It may, however, be possible to exploit portions of the complicated algorithm and factor them down in to a product of smaller and simpler local functions, which are only dependent on a smaller subset of the variables. This factorisation process can be represented visually using a factor graph that shows which variables are arguments of each local function [78]. Each of the nodes, and messages passed between them, can be represented with particle distributions and lead to the use of particle BP, so that problems containing non-linear, non-Gaussian models can be addressed directly.

In order to arrive at an appropriate formulation for the MTT problem, first consider the general case where we wish to estimate a number of parameter vectors $\mathbf{x}_n, n \in \{1, \ldots, N\}$ from a set of measurements $\mathbf{Z}$. The majority of traditional Bayesian methods for estimating the parameter vectors rely on obtaining the posterior pdfs $f(\mathbf{x}_n|\mathbf{Z})$, which are marginals of the joint posterior pdf $f(\mathbf{X}|\mathbf{Z})$ and where $\mathbf{X} = \left[\mathbf{x}_1^T, \ldots, \mathbf{x}_N^T\right]^T$. It is often very difficult or intractable to perform direct marginalisation of the joint posterior pdf, and it would be much more efficient if the joint would factorise in to a product of M lower-dimensional factors such that

$$f(\mathbf{X}) \propto \prod_{m=1}^{M} \psi_m\left(\mathbf{x}^{(m)}\right), \tag{2.50}$$

where $\psi_m(\cdot)$ is a factor, and each argument $\mathbf{x}^{(m)}$ contains a set of parameter vectors $\mathbf{x}_n$, i.e. the set depending on the probabilistic models. In factor graphs, each variable $\mathbf{x}_n$ is represented with a variable node (typically drawn as circles), and each factor $\psi_m(\cdot)$ is represented with a factor node (typically drawn as squares). A variable node and a factor node are adjacent, or connected by an edge, if the variable is an argument of the corresponding factor.

As an example, consider the factor graph shown in Figure 2.7, which contains two variables $\mathbf{X} = \left[\mathbf{x}_1^T, \mathbf{x}_2^T\right]^T$ and three factors. This graph represents the factorisation $f(\mathbf{X}|\mathbf{Z}) \propto \psi_1(\mathbf{x}_1)\psi_2(\mathbf{x}_1, \mathbf{x}_2)\psi_3(\mathbf{x}_2)$. Each of the factors in the graph may be complex in their own right; this may be because they are high-dimensional or that they

are connected to a large number of variables. Techniques such as stretching or opening [78] can be used to help reduce the dimensionality of each factor. In order to stretch a graph, *latent, or redundant,* variables are added to the graph, which have a dependence on a variable that is already present in the formulation. By having these extra variables included, the factorisation becomes more detailed, as the high-dimensional factors have been opened out, and replaced with many low-dimensional factors. As well as the lower-dimensional factors, the messages being passed between these factors should also now have a lower dimension, hence improving efficiency and scalability, and reducing expressiveness. [22, 77].

In practice, a factor graph will be tailored to each individual application or problem, and hence the resulting graph structures and frameworks are varied. Typical structures include trees, bipartite graphs, and loopy graphs [78]. In an ideal world, it would be preferable to have a tree-structured graph, or cycle-free graphs if possible, as it brings a number of advantages. The first advantage is being able to determine the exact number of steps required to pass all of the messages across the graph; this also leads to the second main advantage of being able to guarantee convergence to the true marginals of the joint pdf. Tree-structured graphs provide the exact marginals, whereas for loopy graphs, *this may not be true.*

Performing inference-based tasks on loopy, or cyclic, graphs is far more challenging. When considering these iterative algorithms or processes, we lose the guarantee of beliefs converging; these are now only *approximations of the marginals* of the joint pdf. Because of the loops in the graph, the algorithm changes from having an exact number of steps to pass all messages to an iterative algorithm with no natural stopping point. Although a physical clock may not be required, it is assumed that the passing of messages is synchronised, with one message being passed along an edge in a given direction at a single time-step. This is the *serial, or two-way, schedule.* It is down to a designer to arrange the message passing schedule appropriately; it is clear to say that there could be many different forms of schedule available in practice. Another popular example is the flooding schedule [115, 116]. The *flooding schedule* differs from the two-way schedule, as nodes will not necessarily wait for all other messages to reach their destination before another message is sent along an edge of the graph. Messages will continue to be passed between nodes until a designed stopping criteria has been met. A number of applications have designed appropriate stopping criteria based on convergence data, a chosen number of iterations, or external flags [77, 117, 118]. The MP algorithms used in this work that require stopping criteria will run for a chosen number of iterations.

It is possible to convert these cyclic graphs in to the preferred tree-structured graphs, using algorithms such as the Hugin algorithm, or the Shafer-Shenoy [119] algorithm which are variations of the junction-tree algorithm [118]. The main idea behind these algorithms is to eliminate any loops in the graph and attempt to

cluster them in to single nodes instead; almost the reverse of the "stretching" process described previously. These algorithms are described and thoroughly compared in [120].

2.4.1 Applying BP to Factor Graphs

Messages need to be calculated in each node of the graph and passed to those adjacent nodes that are connected through the edges. All of this is achieved by applying the SPA to the graph. Messages that are entering or leaving a variable node are functions of the associated variable. First, let γ_m be the set of indices n of all of the variable nodes "$\mathbf{x}_n$" that are adjacent to the factor node ψ_m. Secondly, let τ_n be the set of indices m of all of the factor nodes ψ_m that are adjacent to the variable node "$\mathbf{x}_n$". These definitions lead to the two main equations required for passing messages across the graph. The message passed from the factor node ψ_m to an adjacent variable node $\mathbf{x}_n, n \in \gamma_m$ is given by

$$\zeta_{\psi_m \to \mathbf{x}_n}(\mathbf{x}_n) = \int \psi_m(\mathbf{x}^{(m)}) \prod_{n' \in \gamma_n \setminus \{n\}} \xi_{\mathbf{x}_{n'} \to \psi_m}(\mathbf{x}_{n'}) \mathrm{d}\mathbf{x}_{\sim n}, \tag{2.51}$$

where $\int \ldots \mathrm{d}\mathbf{x}_{\sim n}$ is an integration with respect to all vectors $\mathbf{x}_{n'}, n' \in \gamma_m$ except for $\mathbf{x}_n$. The message $\xi_{\mathbf{x}_n \to \psi_m}(\mathbf{x}_n)$ passed from the variable node $\mathbf{x}_n$ to an adjacent factor node $\psi_m, m \in \tau_n$ is given by

$$\xi_{\mathbf{x}_n \to \psi_m}(\mathbf{x}_n) = \prod_{m' \in \tau_n \setminus \{m\}} \zeta_{\psi_{m'} \to \mathbf{x}_n}(\mathbf{x}_n). \tag{2.52}$$

Depending on the chosen message passing schedule, and the factor graph structure, it may take a number of iterations for all appropriate messages to be calculated and passed. After all messages are passed, a belief $\tilde{f}(\mathbf{x}_n)$ is calculated for each of the variable nodes with

$$\tilde{f}(\mathbf{x}_n) = C_n \prod_{m \in \tau_n} \zeta_{\psi_m \to \mathbf{x}_n}(\mathbf{x}_n), \tag{2.53}$$

where C_n is a normalisation constant so that $\int \tilde{f}(\mathbf{x}_n) \mathrm{d}\mathbf{x}_n = 1$ [23].

2.5 Sensor Registration Methods

A large part of this thesis looks at the sensor registration problem. Depending on how the sensor fusion architecture has been defined, be that centralised fusion or distributed fusion, the approach to estimating the sensor registration parameters may change. Sensor registration could incorporate many different parameters such as location, orientation, timing, and intrinsic, sensor-specific factors. For this application, the main focus will be on sensor orientation and physical parameters.

With recent advances in processing power and the extensive capability of Graphics Processing Units (GPUs), machine learning, and deep learning in particular have had a major revival in the literature. Many problems now have neural network-based solutions available; sensor registration included. Early works in machine learning for registration [37] used relatively shallow networks with only a small number of layers. The main idea was to teach the network to try and align the biased measurement data from one sensor with the data from a reference sensor, using gradient descent learning, while modifying the learning rate of the network. This work however, does not address the underlying tracking and fusion problems post-registration. A more recent solution that proposes a deep learning approach [38] has proven successful in the autonomous vehicles application, where a scanning lidar sensor is calibrated with a monocular camera. This more expansive network is trained to perform feature extraction, feature matching and a regression of the parameters. There are however, major drawbacks for using machine learning methods, such as the amount of *realistic* training data required in order for the network to start producing sensible results; an issue especially found in defence and surveillance applications [39, 40].

When considering sensor networks that operate over a larger geographical area, it may be more appropriate to consider a distributed fusion architecture, rather than a centralised system. A full discussion of centralised and distributed fusion architectures is presented in Chapter 3. The work presented in [121, 122] uses distributed fusion on the target tracks generated in each individual sensor. These tracks are generated using RFS MTT methods (Section 2.3.3), and then used as a part of a BP method for registration. Tracks are shared with other neighbouring sensors in the network, along with either a dual-term or a quad-term pseudo-likelihood that encapsulates information about each sensor's location. By trading this information across the sensors, it is shown to be possible to self-localise, or self-calibrate, the network in a pair-wise fashion. A paper by the same authors [123] has shown that this method was successfully applied to a real network containing radar and lidar sensors based on the Sensing for Asset Protection with Integrated Electronic Networked Technology (SAPIENT) architecture [124]. One assumption which could potentially be a drawback to this method is that each sensor should, on its own, be able to produce sensible and accurate tracks. This is a suitable assumption to make when considering a sensor network that only contains sensors which can provide measurements with full position information (eg. radar). This however may not be possible if we were to consider bearings-only sensors such as the camera/infrared systems presented in this thesis. Because these types of sensor operate passively (see Section 2.1), full position information is not available, making the output less accurate and more uncertain.

The method presented in [34] uses unscented filtering techniques, along with a model of the uncertainty over the sensor parameters so that each sensor can

self-localise. Although only a framework for resolving the problem is defined and no results given, the authors conclude that the chosen data association algorithm may have an effect on the overall results; an idea that is also postulated in [14]. This article presents a method for performing joint data association, fusion and registration using the Expectation-Maximisation (EM) algorithm with a Kalman filter-based approach. The results show that attempting to solve the problem jointly, rather than attempting to solve each problem separately, improves the accuracy of the sensor registration.

A lot of previous work has been carried out on solutions to the registration problem which involve creating pseudomeasurements in decentralised or distributed fusion systems. Early works in [16,17] proposed methods that manipulated the state estimates of each of the local sensors such that they gave pseudo-measurements of the sensor biases, with additive zero-mean white noise, and whose covariance was easily calculated. Although these articles touch on the potential asynchronicity of sensors in practice, the angle biases assumed in these works are in the order of tenths of degrees; an order of magnitude smaller than those assumed in this thesis.

A similar pseudomeasurement approach is then extended by Huang et al. in [15] which incorporates an EM algorithm into the estimation process. The result showed improvement over an Extended Kalman filtering based approach, but at the expense of extra computing resources. The batch EM method that is proposed in the article gives highly accurate estimation of the sensor registration parameters, but is restricted to offline implementation cases as it requires all of the data across the scenario. The proposed solution is also based on a track-level fusion architecture, rather than the centralised architecture used in this thesis. A joint EM-KF approach is proposed by Li et al. in [14] which exploits the associations between measurements and targets to estimate the sensor registration parameters. This implementation is presented for a fixed and known number of targets, which is typically not the case in real-world applications.

A more recent approach in [13] proposed to reconstruct the Kalman gains of each of the local trackers at the fusion centre, rather than using extra bandwidth in the system to send this information with the track estimates. A very similar method was then applied to the well-known bearings-only tracking problem in [12]. The method that is applied in both of these articles is presented for the synchronous sensing case, and again, the assumed angle biases that used in the simulated results appear very small compared to those often found in practice.

In summary, the articles listed in Table 2.2 each have gaps or drawbacks in their proposed solutions. The main issues relate to the assumptions that their solutions are built on. This could be the magnitude of the biases being assumed to be very small, or that sensors always operate synchronously; this is definitely not the case in practice. Although machine learning-based solutions are beginning to gain traction

in the defence industry, we still lack the significant amount of realistic data that will be required to train a robust and efficient network. This thesis will look to address all of the features listed in this table and propose a robust solution to the sensor registration and fusion problems.

2.6 Performance Evaluation

In order to assess the performance, and make suitable comparisons between different algorithms, we require some robust metrics that can describe the error between the true target/sensor state(s), and the estimated target/sensor state(s). The sensor registration estimation problem could be considered as a "single-target" problem, and the Root Mean Square Error (RMSE) (Section 2.6.1) would be a suitable metric for comparing its performance. When considering the MTT estimation problem, defining the error is less straight-forward. Because we are attempting to track multiple targets, differences in cardinality must also be taken account of; not just the localisation error. The Optimal Subpattern Assignment (OSPA) and Generalised Optimal Subpattern Assignment (GOSPA) metrics are suitable for this and are introduced in Section 2.6.2 and Section 2.6.3, respectively. One other metric that will be introduced here is Averaged Normalised Estimation Error Squared (ANEES), which could be used in practice to determine the credibility of estimates being generated by a filter; this is introduced in Section 2.6.4.

2.6.1 Root Mean Square Error (RMSE)

The RMSE is a well-known and widely used metric for estimation problems [21,125]. Let us assume that $\mathbf{a}$ represents the true state vector, and $\mathbf{b}$ represents the estimated state vector. With that, the RMSE can be defined as

$$\text{RMSE}(\mathbf{b}) = \sqrt{\text{MSE}(\mathbf{b})} \tag{2.54}$$

where $\text{MSE}(\mathbf{b}) = ||\mathbf{a} - \mathbf{b}||_2^2$ is the mean square error, so that

$$\text{RMSE}(\mathbf{b}) = ||\mathbf{a} - \mathbf{b}||_2. \tag{2.55}$$

Because we are looking at time series data, and averaging over a number of MC trials, the RMSE will be computed using

$$\text{RMSE}(\mathbf{b}) = \sqrt{\frac{\sum_{t=1}^{T}(\mathbf{a}_t - \mathbf{b}_t)^2}{T}}. \tag{2.56}$$

The RMSE will be used in Chapter 5 and in Chapter 6 when directly comparing the performance of the two differing methods for joint registration and fusion.

Table 2.2: Sensor Registration Summary

Article	Meas. Fusion	Asynchronous	Multi-Target	Heterogeneous	Realistic Assumptions	Practical Solution
Karniely et al. [37]	✗	✗	✗	✗	✗	✗
Schneider et al. [38]	✓	✗	✗	✓	✗	✗
Uney et al. [121]	✗	✓	✓	✓	✗	✓
Vermaak et al. [34]	✓	Unknown	Unknown	Unknown	Unknown	✓
Lin et al. [16]	✗	✓	✓	✗	✗	✓
Huang et al. [15]	✗	✗	✓	✓	✗	✗
Li et al. [14]	✓	✗	✓	✗	✗	✗
Taghavi et al. [13]	✓	✗	✓	✓	✗	✓

2.6.2 OSPA Metric

The Optimal Subpattern Assignment metric [126] is a widely used benchmarking tool for comparing accuracies of MTT algorithms. The metric follows on from the Optimal Mass Transfer (OMAT) metric [127] and overcomes a number of issues that restrict its use in practice, such as consistency in results, geometry dependent behaviour, and the zero-cardinality problem. The OSPA distance is made up of a cardinality error and a localisation error between two sets $\mathbf{X}$ and $\mathbf{Y}$ with cardinalities m and n (it is assumed that $\mathbf{X}$ has at most as many elements as $\mathbf{Y}$). The distance is defined by [126, Eq. (3)]

$$d_p^{(c)}(\mathbf{X},\mathbf{Y}) = \left[\frac{1}{n}\min_{\pi\in\prod_n}\sum_{i=1}^{m} d^{(c)}(\mathbf{x}_i,\mathbf{y}_{\pi(i)})^p + c^p(n-m)\right]^{\frac{1}{p}}, \tag{2.57}$$

which includes a chosen order parameter p and a cut-off distance c. The distance function $d^{(c)}(\mathbf{x},\mathbf{y}) = \min(c, d(\mathbf{x},\mathbf{y}))$ is an appropriate distance measure such as the Euclidean distance, cut-off at c. $\prod_n$ denotes the set of all possible permutations of the numbers $1,\ldots,n$. The OSPA metric can be broken down in to its two component parts, the localisation error and the cardinality error such that

$$e_{p,\mathrm{loc}}^{(c)}(\mathbf{X},\mathbf{Y}) = \left[\frac{1}{n}\cdot\min_{\pi\in\prod_n}\sum_{i=1}^{m} d^{(c)}(\mathbf{x}_i,\mathbf{y}_{\pi(i)})^p\right]^{\frac{1}{p}}, \tag{2.58}$$

$$e_{p,\mathrm{card}}^{(c)}(\mathbf{X},\mathbf{Y}) = \left[\frac{c^p(n-m)}{n}\right]^{\frac{1}{p}} \tag{2.59}$$

The order parameter is used in a similar way as in other MTT metrics. As the value of p increases, the metric becomes harsher on estimates that are not close to any of the true objects. When $p = 1$, the sum of the localisation and cardinality components of the metric equals the total metric. However in practice, $p = 2$ is a more realistic and preferred choice, as it will result in smoother distance curves and is consistent with other metrics that use a p-th order average construction. The cut-off distance c defines a weighting between penalising cardinality errors as opposed to localisation errors. The OSPA metric is exploited in Chapter 4 to compare tracking accuracy in the range of scenarios presented.

The metric itself can be computed with the help of a data association algorithm from Section 2.3.1, in order to find the optimal assignment of two sets of points. Algorithms such as Munkres are more often found inside the tracking algorithm itself, but these techniques are also applied to metrics such as OSPA to find the minimum distance between the truth data and the generated estimates.

2.6.3 GOSPA Metric

Although the OSPA metric was designed to overcome a number of issues with previous metrics for MTT, there are still a number of limitations that remain. One example is that OSPA does not encourage the algorithms to have the smallest number of missed or false targets as possible. A more desirable metric would include a localisation error for properly detected targets, and for false and missed targets, rather than a localisation error for the targets in the smallest assigned set plus a cardinality mismatch penalty. The Generalised Optimal Subpattern Assignment (GOSPA) metric [128] is an important advancement to the OSPA metric which is able to penalise localisation errors for detected targets, false targets and missed targets.

The GOSPA metric is defined as

$$d_p^{(c,\alpha)}(\mathbf{X},\mathbf{Y}) \triangleq \left[\min_{\pi\in\prod_n}\sum_{i=1}^{m} d^{(c)}(\mathbf{x}_i,\mathbf{y}_{\pi(i)})^p + \frac{c^p}{\alpha}(n-m)\right]^{\frac{1}{p}} \tag{2.60}$$

with p and c having the same meaning as with the OSPA metric, and the inclusion of an additional parameter α, which gives a choice of cardinality mismatch cost. In practice, this should be set to $\alpha = 2$, since intuitively, the cost for a single unassigned (false or missed) target should be the same, whether or not the target can be associated to another target in the permutation in (2.60). The GOSPA metric is used in both Chapter 5 and Chapter 6 in order to compare the differing methods in terms of their target tracking accuracy.

2.6.4 ANEES Metric

Although it is not discussed often in the MTT literature, the Averaged Normalised Estimation Error Squared (ANEES) metric [129, 130] should be highlighted as an important metric for determining the credibility of a state estimate produced by a tracking algorithm. There are a plethora of different techniques available for assessing estimation accuracy and performance (RMSE, OSPA and GOSPA being a small subset of them). These metrics work by comparing the filter estimates to some form of reference data, but how do we know that the estimates coming from the filter are reliable or credible? The ANEES score will tell us if the filter is being overly-confident in the estimate it is providing. The ANEES can be computed by

$$ANEES = \frac{1}{nN}\sum_{i=1}^{N}(\mathbf{x}_k - \hat{\mathbf{x}_k})^T\mathbf{P}_k^{-1}(\mathbf{x}_k - \hat{\mathbf{x}_k}) \tag{2.61}$$

where $(\mathbf{x}_k - \hat{\mathbf{x}_k})$ is the error in the state estimate, $\mathbf{P}_k$ is the covariance matrix associated with the state, n is the dimension of the state, and N is the number of runs in the Monte-Carlo test [130]. If the state estimation error and the covariance are the same, the ANEES score should be 1, and we can say the filter is credible.

2.7 Conclusions

There are two main conclusions that can be drawn from this chapter; firstly, there is already a plethora of different techniques available for resolving sensor registration issues. However, these methods are based on strong assumptions that do not hold in practice:

The magnitude of the angle biases presented in previous works appear to be very small, in the order of tenths, or even hundredths, of degrees.

Synchronous sensor operation is an important factor that has been shown in simulated scenarios in previous works. This is almost never achievable in practical systems, especially when commercial-off-the-shelf (COTS) sensors are considered.

Offline estimation of parameters is critical for finding an initial set of sensor registration parameters. However, online estimation schemes such as the proposed methods used later in this thesis, are required for cases where the registration parameters are dynamic and time-varying.

Secondly, a number of the algorithms that have been presented can be adapted or combined using appropriate frameworks to create new solutions for resolving the sensor registration problem, which remove the aforementioned assumptions.

This chapter has provided an overview of a number of key concepts and techniques from the previous literature in the areas of target tracking, sensor fusion and sensor registration, which should put the contents of this thesis into context and clearly outline the novelty of the work that follows. It should also provide enough background material such that someone who is not familiar with tracking and fusion methods can understand the content of the remaining chapters.

Chapter 3

Modelling and Assumptions

With the background material presented previously in Chapter 2 in mind, the aim of this chapter is to provide a summary of underlying assumptions that the work presented in later chapters is based on, and to present a number of algorithm implementation details. This chapter will include information on topics such as:

- the sensor fusion architecture that has been simulated;
- any underlying assumptions that have been made in terms of the tracking scenarios, the sensor fusion architecture and the proposed solutions.
- the derivation of appropriate Jacobian matrices to overcome the non-linearity between the state space and observation space;
- the underlying dynamic, and observation models that will form the MTT algorithms.

3.1 Sensor Fusion Architecture

There are a range of sensor fusion architectures available for implementation, each with their own advantages and disadvantages. From the literature, there appears to be no consensus or agreement as to which type of architecture is best, and it is dependent on the application area or scenario. As the number of sensors in a network increases, issues surrounding *scalability* become more apparent, and these issues could lead to a degradation in tracking and fusion performance.

3.1.1 Scalability Problems

There are quite a number of problems that arise in practical multi-sensor systems which are often overlooked, or avoided in parts of the data fusion literature. They are often treated as separate problems, but they must all be considered when attempting to deploy real-world systems. This section will first introduce further issues that

become more prominent when the number of sensors contributing to the data fusion process increases.

Latency and Delays

As the size of the surveillance region, and the distances between sensor locations increases, we begin to see long-haul links between the remote sensors, and the centralised Fusion Centre (FC) [131]. Sensors may either send out their time-stamped measurement data, or their time-stamped state estimates to this fusion centre for further processing. Consider two differing scenarios; firstly, a small remote region that needs widespread surveillance coverage, using low Size Weight and Power (SWaP) sensing equipment. Each individual sensor may not have the capability or power to perform complex MTT and fusion routines, and therefore needs to transmit this information to the fusion centre. A second appropriate scenario may be in the airborne surveillance case, where fast moving platforms may need to relay measurement or track data back to a ground station many tens of kilometres away. Because of the long distances that may need to be covered, or because of the low Signal-to-Noise Ratio (SNR) by the time the information arrives, critical measurement data may either be delayed, or lost entirely. With data missing, the fusion system may not be able to reach it's own "optimal" performance, and tracking accuracy could become diminished. The missing data problem can be partially solved using retransmission and retrodiction techniques such as the one presented in [132].

The latency problem, where the information still arrives but after an unknown and potentially varying time delay can often be less than trivial to overcome. Different sensors may experience different lengths of latency at each transmission, and at the fusion centre, these measurements affected by latency may be interleaved with measurements that are subjected to little or no latency. Methods for resolving the Out-of-Sequence Measurement (OOSM) problem have been illustrated in the literature [21, 133]. The goal of these algorithms is to efficiently update the current state estimates with measurements generated much earlier than the current time, while avoiding the need to reorder any measurements or reassessing any past data association (DA) decisions. The DA problem in this case could be very large, as we may need to redo a very large number of measurement-to-track associations, depending on how long the delay has been. This large association problem was reviewed, and different solutions compared, in [134].

Network Architectures

With the sensor network size increasing, the amount of data being transferred between sensors and the FC will also become much larger. This could put excessive strains and loads on the communication links and further contribute to the latency issues discussed previously. One aspect that could be adapted is the *fusion net-*

work architecture. In this work, the centralised architecture will be considered, where measurement-level fusion is performed using all available sensor measurements. Centralised fusion at the measurement level should be the "optimal" method for performing sensor fusion, and is best suited for cases where a single platform with multiple sensors is considered, or in situations where sensors may be spread out over a small geographical area, for example at an airport or a harbour, and physical connections to the FC are available at all times. Each sensor would, in practice, perform some pre-processing of the raw data to generate these measurements, be that a simple thresholding step, or more advanced image processing techniques to detect targets. In order to save on computation however, an alternative could be to consider a *decentralised, or distributed* architecture. This would mean that each individual sensor would be responsible for performing it's own target tracking routine, and only track estimates would need to be shared, vastly reducing the amount of data that needs to be transmitted. The change to a distributed system comes at a cost however; the tracking accuracy becomes "sub-optimal" [21, 93].

A further alternative could be to consider a *hybrid fusion architecture*, which contains a combination of both centralised and decentralised architectures. This may be appropriate for the following exemplar scenario. Consider a multi-platform case containing a number of airborne platforms that are attempting to track multiple targets on the ground. Each of these platforms is likely to contain a multi-modal suite of sensors available, improving the probability of detecting targets. We can now imagine that this fusion problem could almost be broken down in to a hierarchical structure of fusion "layers". The content and ordering of these layers could also take multiple forms; for example we could -

1. Perform measurement-level fusion on each individual platform using an on-board FC; perform track-level fusion between the platforms using a FC on the ground. This would be implemented much like the structure shown in Figure 3.3, with the Track Fusion algorithm(s) hosted on a ground platform, or central computing centre.

2. Perform measurement-level fusion on each individual platform using an on-board FC; perform track-level fusion between the platforms. This again would be implemented much like the structure shown in Figure 3.3, however the track fusion processes would either be hosted on one of the airborne platforms, or potentially in some form of cloud-based solution. This could also be seen as a form of small airborne mesh network.

3. Perform track-level fusion between each of the sensors on each platform and then perform track-level fusion between the platforms. In this case, tracks would be generated inside each sensor as shown in Figure 3.2, with the Track Fusion algorithm(s) hosted on one of the platforms or in the cloud.

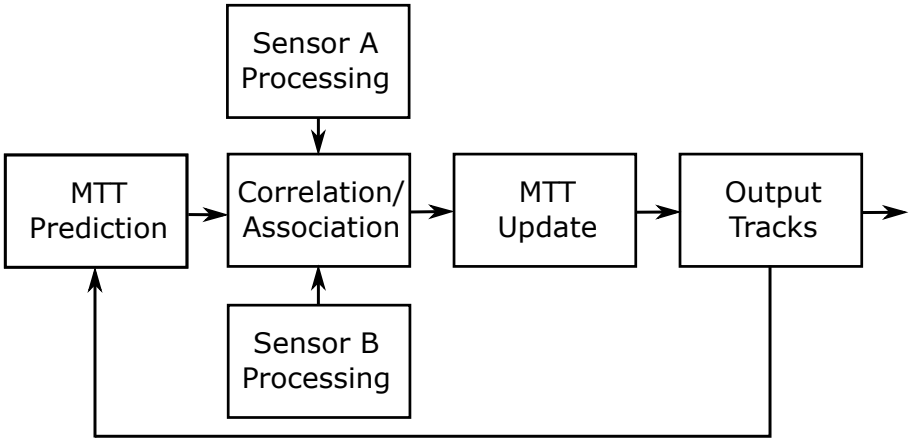


Figure 3.1: Measurement-level Fusion.

All of these possibilities would lead to different output tracks, depending on which fusion architecture we adopt and which order we choose for sensors to be used.

Ordering of Sensors

When considering a larger set of sensors in a synchronous network where sensors record measurements at the same time, deciding on a process to perform sensor updates is an important design choice. There are two main choices available; a batch processing (multi-sensor) update, or an iterative (iterated-corrector) update through each sensor. In general, the batch processing method does not scale well in the number of sensors, and the number of measurements per sensor, because of the high-dimensional association problem that needs to be resolved. For example, if we were to have a sensor network containing five sensors observing a congested scene with 100 targets, this would generate 500 different detections (assuming the detection profile is perfect with no false alarms, this would only make the problem even larger). Trying to use an association algorithm such as Munkres on this scale of problem would be rather time-consuming.

The alternative to performing a series of updates from each of the sensors provides a suitable alternative to the batch processing method, and can often be performed more efficiently in practice [114]. Choosing which order to perform sensor updates in can give quite vastly differing output tracks and resultant accuracies. For situations where we have high probabilities of detection in all sensors, the effect of ordering is negligible in most cases. However when we have fluctuating cases, or sensors with low probabilities, ordering is critical. Problems with ordering have been presented in [114, 135, 136] previously. If a sensor with a low probability of detection or higher measurement uncertainty is placed last in the "queue" for updates, then we are likely to see worse tracking accuracy. In order to gain the best accuracy from the iterative scheme, the worst sensor measurements should be updated with first, and then use the measurements that get progressively more accurate.

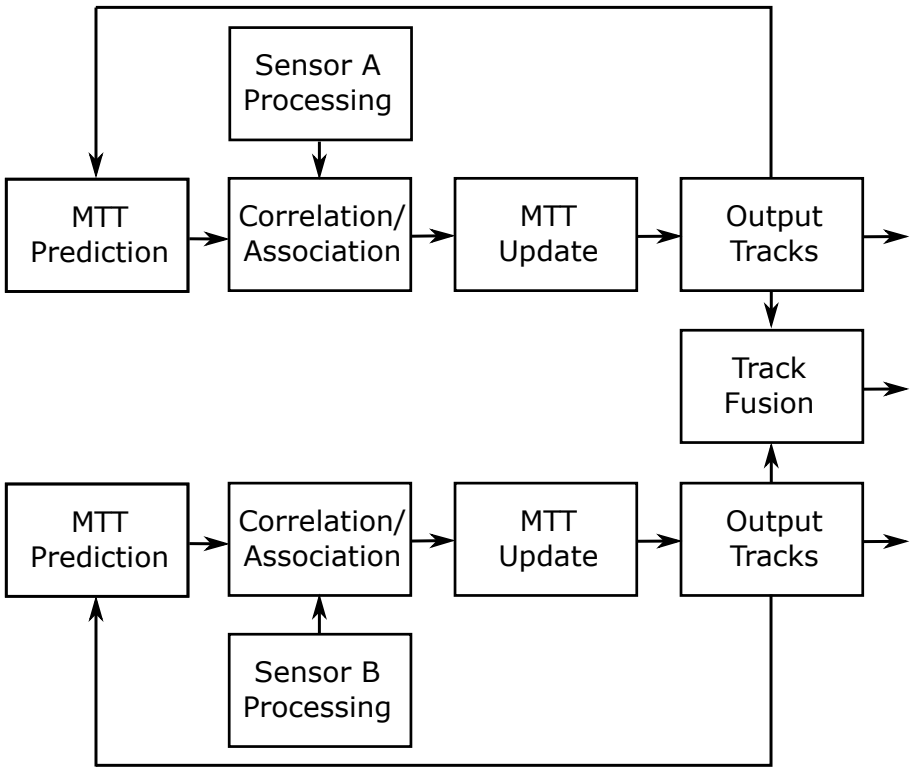


Figure 3.2: Track-level Fusion.

3.2 Proposed Method Framework

An overview of the proposed framework for resolving both the sensor registration and fusion problems, which is exploited in the remaining chapters, is given in Algorithm 1, with a graphical representation in Figure 3.4. In terms of how the main algorithm operates, it closely follows the traditional tracking algorithm routine of a prediction step and an update step. The routine is initialised with a set of particles $i = 1, \ldots, N$ which represent the different sensor registration states q^i_{k-1}, and each particle has an associated weight w^i_{k-1}. Each of these particles also effectively contains a realisation of the underlying multi-target state θ^i_{k-1}.

During the prediction step, for each particle that we have in the mixture, we will adjust the weight of each particle using some underlying model. Note that in Algorithm 1 we still assume that the sensor registration states are static and fixed and hence are not predicted. We then perform the prediction step for each of the realisations of the multi-target state in the same way that a normal MTT filter would, using a NCV model for example.

In the update step, we again need to perform updates on both the multi-target state realisations, and the sensor registration states. Firstly we perform the multi-target state update as normal, however during this process we will compute the Multi-Object Likelihood (MOL) calculation in each of the updates. This MOL value is required to then update the sensor registration particle weights using (3.2). We will only update the registration weights if the most recent measurements have

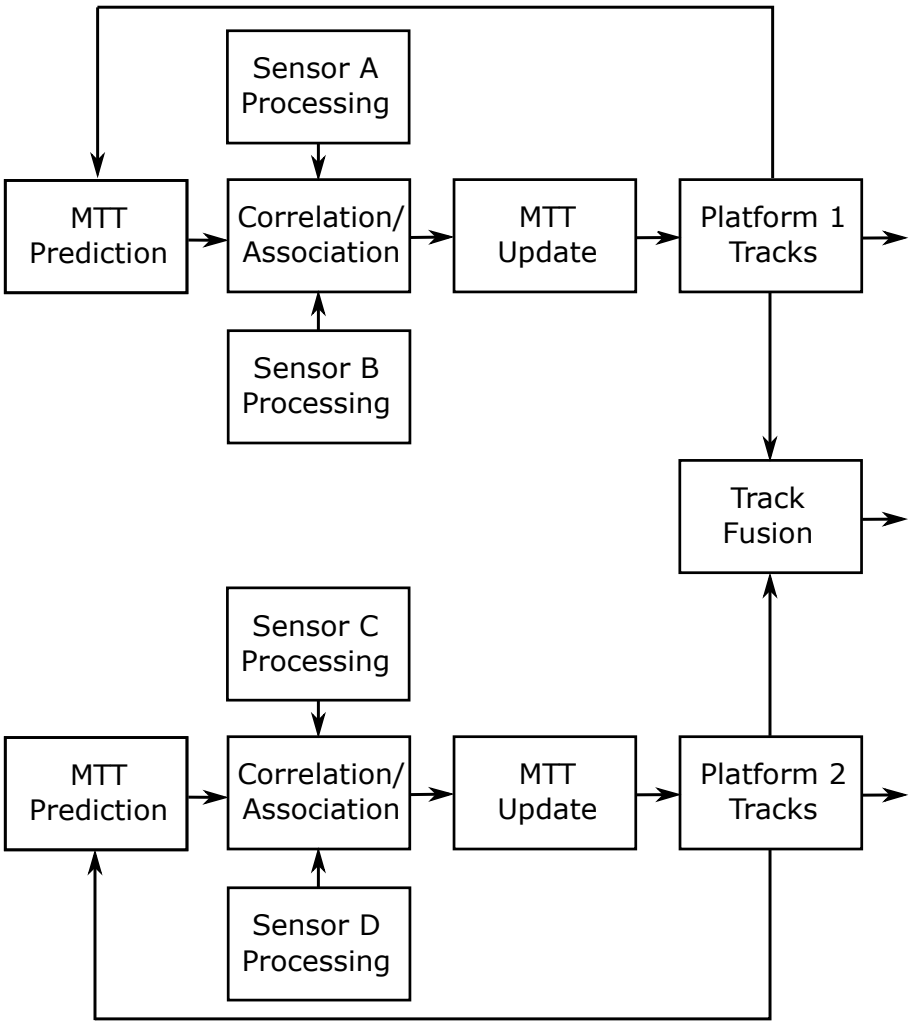


Figure 3.3: Hybrid Fusion.

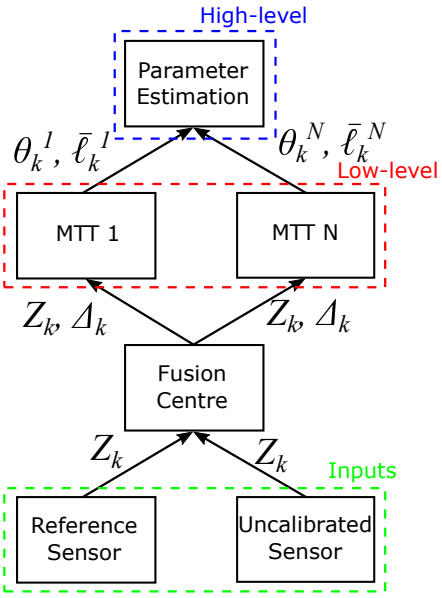


Figure 3.4: Flowchart of the sensor fusion architecture, and the proposed method.

come from a sensor that we assume is not registered correctly.

An introduction to this framework, *the single-cluster method,* is provided in Chapter 2. As eluded to in this introduction to the single-cluster method, the main recursion can be broken down into two separate recursions, one for the parent process, and one for the offspring process. Firstly for the parent process, denoted with $\hat{\cdot}$:

$$\hat{P}_{k|k-1}(\mathbf{q}_k|\mathbf{Z}_{1:k-1}) = \int \hat{f}_{k|k-1}(\mathbf{q}_k|\mathbf{q}_{k-1})\hat{P}_{k-1}(\mathbf{q}_{k-1}|\mathbf{Z}_{1:k-1})\mathrm{d}\mathbf{q}_{k-1}, \tag{3.1}$$

$$\hat{P}_k(\mathbf{q}_k|\mathbf{Z}_{1:k}) = \frac{\hat{\ell}_k(\mathbf{q}_k|\mathbf{Z}_k)\hat{P}_{k|k-1}(\mathbf{q}_k|\mathbf{Z}_{1:k-1})}{\int \hat{\ell}_k(\mathbf{q}'|\mathbf{Z}_k)\hat{P}_{k|k-1}(\mathbf{q}'|\mathbf{Z}_{1:k-1})\mathrm{d}\mathbf{q}'}, \tag{3.2}$$

where $\hat{f}_{k|k-1}(\mathbf{q}_k|\mathbf{q}_{k-1})$ is a first-order Markov model that represents the change in sensor registration parameters over time and $\hat{\ell}_k(\mathbf{q}_k|\mathbf{Z}_k)$ is a MOL function. The recursion for the offspring process is:

$$P_{k|k-1}(\mathbf{X}_k|\mathbf{q}_k, \mathbf{Z}_{1:k-1}) = \int f_{k|k-1}(\mathbf{X}_k|\mathbf{X}_{k-1})P_{k-1}(\mathbf{X}_{k-1}|\mathbf{q}_k, \mathbf{Z}_{1:k-1})\mathrm{d}\mathbf{X}_{k-1} \tag{3.3}$$

$$P_k(\mathbf{X}_k|\mathbf{q}_k, \mathbf{Z}_{1:k}) = \frac{\ell_k(\mathbf{X}_k|\mathbf{q}_k, \mathbf{Z}_k)P_{k|k-1}(\mathbf{X}_k|\mathbf{q}_k, \mathbf{Z}_{1:k-1})}{\int \ell_k(\mathbf{X}'|\mathbf{q}_k, \mathbf{Z}_k)P_{k|k-1}(\mathbf{X}'|\mathbf{q}_k, \mathbf{Z}_{1:k-1})\mathrm{d}\mathbf{X}'} \tag{3.4}$$

where $f_{k|k-1}(\mathbf{X}_k|\mathbf{X}_{k-1})$ is the chosen dynamical model for the target states, as described in Section 2.1. This recursion follows the typical tracking routine equations shown in Section 2.2.

3.3 Underlying Assumptions

Although an aspect of this thesis is to relax or remove assumptions that have been made in previous articles in sensor registration and fusion, a number of assumptions will still need to be kept in place to keep the problem, and the proposed solutions, tractable. The work that is presented in the following chapters is based on the following assumptions:

1. All simulated target trajectories will follow a *consistent dynamic model*, and multiple model estimation schemes such as IMM will not be required;
2. Each sensor will generate its own measurements *independently* of other sensors;
3. The false alarm distributions for each sensor *are independent*;
4. The connections between the Fusion Centre and the individual sensors are *always available*;
5. There is *no latency* between the sensors and the FC, and OOSMs [21] are not simulated;

Algorithm 1 Joint Sensor Registration and Fusion

Input: Set of particles $\{q^i_{k-1}, w^i_{k-1}, \theta^i_{k-1}\}_{i=1}^N$
Set of measurements $\mathbf{Z}_k$

procedure PREDICTION
  for $1 \leq i \leq N$ **do**
    $w^i_{k|k-1} = \mathsf{ParentPrediction}(w^i_{k-1})$ ▷ Eq. (3.1)
    $\theta^i_{k|k-1} = \mathsf{OffspringPrediction}(\theta^i_{k-1})$ ▷ Eq. (3.3)
  end for
end procedure

procedure UPDATE
  for $1 \leq i \leq N$ **do**
    $\theta^i_k = \mathsf{OffspringUpdate}(\theta^i_{k|k-1}, \mathbf{Z}_k)$ ▷ Eq. (3.4)
    if $\mathbf{Z}_k$ from reference sensor **then**
      $w^i_k = w^i_{k|k-1}$
    else if $\mathbf{Z}_k$ from uncalibrated sensor **then**
      $w^i_k = \mathsf{ParentUpdate}(\theta^i_{k|k-1}, w^i_{k|k-1})$ ▷ Eq. (3.2)
    end if
  end for
end procedure

Output: Set of particles $\{q^i_k, w^i_k, \theta^i_k\}_{i=1}^N$

6. Two or more sensors will *never provide measurements at the same point in time*;

7. Data transmitted between the sensors and the FC *will never be lost, or need to be retransmitted.*

These assumptions are in place to help constrain the problems of registration and fusion, which are the main topics of this thesis. It should be acknowledged that in practice, problems with network connections and latency are likely to be present and should be accounted for accordingly.

3.4 Tracking Model Definitions

This section will introduce the chosen underlying dynamical model, and sensor observation models, that will be applied in the simulations presented in the forthcoming chapters.

3.4.1 Dynamic Model

The centralised fusion centre should contain all of the raw range-bearing measurements from the radar, and the bearing-only measurements from the camera or IRST

system, recorded at a given iteration k which corresponds to a physical time t_k. The offspring process is carried out using a 4-D Cartesian state vector with components

$$\mathbf{x}_k = [x_k \ \dot{x}_k \ y_k \ \dot{y}_k]' \tag{3.5}$$

where x_k, y_k are the x and y positions of a target, and $\dot{x}_k, \dot{y}_k$ are the x and y velocities of a target. As we are dealing with a maritime surveillance application, it will be assumed that each and every target will follow a NCV dynamic model, as defined in Section 2.1 [21, 52], which is described by

$$\mathbf{x}_k = \mathbf{F}_k \mathbf{x}_{k-1} + \mathbf{w}_k \tag{3.6}$$

where $\mathbf{F}_k$ is the state transition matrix

$$\mathbf{F}_k = \begin{bmatrix} 1 & \Delta_k & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta_k \\ 0 & 0 & 0 & 1 \end{bmatrix}, \ \Delta_k = t_k - t_{k-1}, \tag{3.7}$$

t_k is the current physical time, t_{k-1} is the last physical time that the target states were updated, and $\mathbf{w}_k$ represents the zero-mean white Gaussian process noise with covariance

$$\mathbf{Q}_k = \begin{bmatrix} u\Delta_k^3/3 & u\Delta_k^2/2 & 0 & 0 \\ u\Delta_k^2/2 & u\Delta_k & 0 & 0 \\ 0 & 0 & u\Delta_k^3/3 & u\Delta_k^2/2 \\ 0 & 0 & u\Delta_k^2/2 & u\Delta_k \end{bmatrix} \tag{3.8}$$

and u is the acceleration noise value in $m^2 s^{-3}$ in both the X and Y directions.

3.4.2 Observation Models

For the presented sensor setup, two differing observation models are required for the sensors. Firstly for the radar, denoted by superscript R, the measurement model is defined as

$$\mathbf{z}_k^R = h^R(\mathbf{x}_k) + \eta_k^R, \tag{3.9}$$

with,

$$h^R(\mathbf{x}_k) = \begin{bmatrix} \rho_k \\ \phi_k \end{bmatrix} = \begin{bmatrix} \sqrt{x_k^2 + y_k^2} \\ \tan^{-1}(x_k, y_k) \end{bmatrix}, \tag{3.10}$$

where $\rho_k > 0$, $\tan^{-1}(x_k, y_k)$ is the four-quadrant inverse tangent function, and the resulting azimuth measurement ϕ_k lies inside $[0, 2\pi)$, clockwise from North. The additive measurement noise term η_k^R is defined by

$$\eta_k^R \sim \mathcal{N}(\eta_k^R; \mathbf{0}, \mathrm{diag}(\sigma_{\rho_r}^2, \sigma_{\phi_r}^2)) \tag{3.11}$$

where σ_{ρ_r} and σ_{ϕ_r} are the radar's range and azimuth standard deviations respectively.

Secondly, the IRST observation model, denoted as C for camera, is described by

$$\mathbf{z}_k^C = h^C(\mathbf{x}_k) + \eta_k^C \tag{3.12}$$

where

$$h^C(\mathbf{x}_k) = \phi_k = \tan^{-1}(x_k, y_k). \tag{3.13}$$

The additive noise term η_k^C is defined by

$$\eta_k^C \sim \mathcal{N}(\eta_k^C; \mathbf{0}, \sigma_{\phi_c}^2) \tag{3.14}$$

where σ_{ϕ_c} is the IRST azimuth standard deviation. This model would also be appropriate for other types of camera or imaging system that provide angle-only measurements, and as such, will be denoted with a superscript C for camera.

3.5 Jacobian Matrix Derivation

From the introduction to the Extended Kalman Filter (EKF) in Section 2.2.1, in order to overcome the non-linearity between the Cartesian state-space, and the Polar observation space assumed in this work, the tracking filters will need to be implemented with EKF update steps. The Jacobian matrices for this implementation are given by

$$\mathbf{J}_{pq} = \frac{\partial h_p}{\partial \mathbf{x}_q}, \quad p \in \{\rho, \phi\},\, q \in \{1, \ldots, 4\} \tag{3.15}$$

where h_p is the conventional Cartesian to Polar transformation

$$h_\rho = \rho = \sqrt{x^2 + y^2},\ h_\phi = \phi = \tan^{-1}(x, y),$$

$\tan^{-1}(x, y)$ is the four-quadrant inverse tangent function and $\mathbf{x} = [x, \dot{x}, y, \dot{y}]^T$. During an update with radar measurements containing range and azimuth values, the Jacobian matrix used is

$$\mathbf{J}_R = \begin{bmatrix} \frac{x}{\sqrt{x^2+y^2}} & 0 & \frac{y}{\sqrt{x^2+y^2}} & 0 \\ \frac{-y}{x^2+y^2} & 0 & \frac{x}{x^2+y^2} & 0 \end{bmatrix} \tag{3.16}$$

and for an IRST or camera update, which only contains an azimuth value, the Jacobian becomes

$$\mathbf{J}_I = \begin{bmatrix} \frac{-y}{x^2+y^2} & 0 & \frac{x}{x^2+y^2} & 0 \end{bmatrix}. \tag{3.17}$$

3.6 Conclusion

This chapter has set out the main underlying assumptions that will help form the work presented in the forthcoming chapters. It has provided details of the sensor fusion architecture that has been chosen for development, with the reasoning behind its selection, and an overview of alternative fusion frameworks. The information provided on the dynamic model, observation models, and Jacobian matrices will be consistent across the remaining chapters.

Chapter 4

Joint Registration and Fusion with Point Process Methods

This chapter will introduce an implementation of the *single-cluster method* [110,111] that will perform joint sensor registration and fusion between heterogeneous sensors. These sensors have an angular bias between their respective frames of reference. The implementation will also involve a grid-based method [49] for representing the potential sensor registration configurations at each iteration. The results from this chapter will lead in to chapter 5, where some limitations of this implementation are addressed. The work that will be presented in this chapter was published in [35]. The main contributions from this chapter include:

- the application of the single-cluster method, which incorporates both the sensor registration and fusion problems in a joint manner, in contrast to existing techniques in the literature that solve tracking and registration problems separately using pseudo-measurements [12, 13, 15, 16]; the proposed method also avoids the computationally expensive data association problem found in other joint approaches [14];

- a new prediction model for tracking changes in the sensor registration configuration, which is included in the parent process computations; this model also incorporates non-uniform sampling information to overcome the asynchronous aspect of the different sensors;

- non-linear observation models that take account of the sensor measurements being provided in polar coordinates, and the Multi-Target Tracking (MTT) output being provided in Cartesian coordinates; therefore, two implementations of non-linear estimation algorithms are used, including a novel Extended Kalman Filter (EKF) version of the Panjer filter;

- a comprehensive set of simulations and results, involving a typically challenging tracking scenario where target trajectories cross one another; and more

importantly, the simulations consider a much larger angular bias/offset between the sensors, by an order of magnitude, compared to those simulated in [13, 17].

- a set of plausible results based on a real data set taken from a live trial in a maritime environment. Due to issues with the elevation data in the real data collection, it has been necessary to test on two-dimensional, range-azimuth data.

4.1 Implementation

The parent process has been represented with a one-dimensional even spread of particles on a fixed-grid [49], with each particle, at time k, representing a different sensor registration configuration such that $q_k^i = \phi_b$, with $i = 1, \ldots, N$. As eluded to in Section 2.2.2, the grid-based method provides the *optimal filtering recursion* if the state space is discrete and has a finite number of states. In this case, the continuous, wrapped, angle values $[-180° \rightarrow 180°]$ have been truncated so that the limits of the angle values are now $[-10° \rightarrow 10°]$. This smaller space is then discretised into suitable angle increments, allowing for the grid-based estimation method to be applied here. This then presents a trade-off between resolution and computation time; a higher resolution grid containing more samples would require a larger computation time to evaluate. The chosen resolution for the grid in this work is $0.1°$ increments, giving $N = 201$ discrete samples to evaluate.

Over time, the registration hypotheses stay constant, removing the need for a particle resampling step such as those in [65]. This leads to a simpler recursion for predicting and updating the weights on the grid. At the moment, only one registration parameter will be estimated in the model. With the particles being fixed to a grid, the particle weights w_k^i can be computed using the grid-based methods in Chapter 2. The implemented recursion for predicting and updating the parent process weights is

$$w_{k|k-1}^i = \sum_{j=1}^{N} w_{k-1}^j \hat{f}_{k|k-1}\left(q_k^i | q_k^j\right) \tag{4.1}$$

$$w_k^i = \frac{w_{k|k-1}^i \hat{\ell}_k(q_k^i | \mathbf{Z}_k)}{\sum_{j=1}^{N} w_{k|k-1}^j \hat{\ell}_k(q_k^j | \mathbf{Z}_k)} \tag{4.2}$$

where $\hat{f}_{k|k-1}\left(q_k^i | q_k^j\right) = \hat{f}_{k|k-1}^{i-j}$ is a pre-defined, discrete density over the difference in registration indices, and $\hat{\ell}_k(q_k^i | \mathbf{Z}_k)$ is the Multi-Object Likelihood (MOL) function evaluated for a given sensor registration configuration q_k^i. Equation (4.1) can be

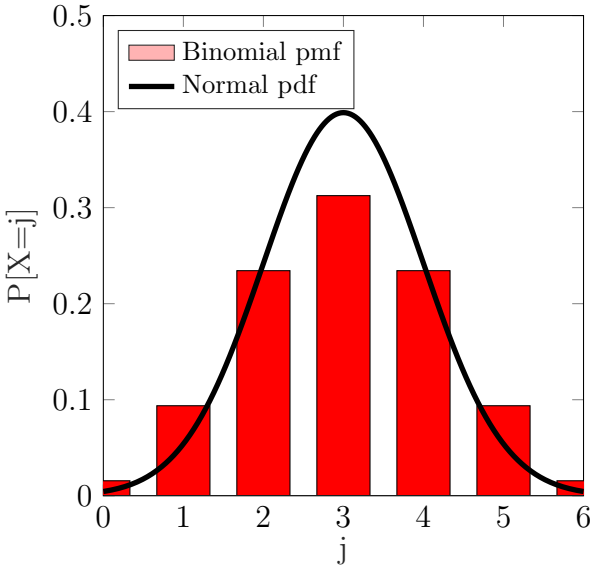


Figure 4.1: Binomial approximation to a normal pdf.

seen as a convolution between the weights w_{k-1}^{j} with the chosen kernel $\hat{f}_{k|k-1}^{i-j}$:

$$w_{k|k-1}^{i} = \hat{f}_{k|k-1}^{i-j} * w_{k-1}^{j}. \tag{4.3}$$

In this work, the transition function is modelled as a perturbation with a discretised, wrapped Gaussian distribution. The new integer index is the existing integer index plus the perturbation generated from this distribution. This distribution can be approximated using a finite-support, shifted binomial distribution where $u \sim B(n, p)$ and $i = j + u - n/2$ is the relationship between the predicted registration index i, and the particle index j. The binomial distribution needs to be shifted by $n/2$ so that it can also model a negative increment; without this, it would appear that the registration error would always be non-negative. Here, the values $n = 6$ and $p = 0.5$ are used, such that Equation (4.1) can be calculated as the convolution of w_{k-1} with the kernel $B(6, 0.5) = \{0.0156; 0.0938; 0.2344; 0.3125; 0.2344; 0.0938; 0.0156\}$. This kernel, and its approximation to a normal probability density function (pdf) is shown in Figure 4.1. The choice of the binomial distribution in this case is justified as the grid is made up of discrete samples, and the binomial distribution is discrete. An alternative approximation for the transition kernel could come from the von Mises distribution; a continuous distribution on the circle [137].

The parent process weights are initialised to a flat prior distribution such that $w_0^i = 1/N$, where all sensor registration configurations are equally likely. Depending on which filter is chosen to perform the offspring process, the information contained inside θ_k^i will be different. For the Probability Hypothesis Density (PHD) filter, this will contain the intensity μ_k of the multi-target distribution, or both μ_k and the variance in the case of the Panjer filter.

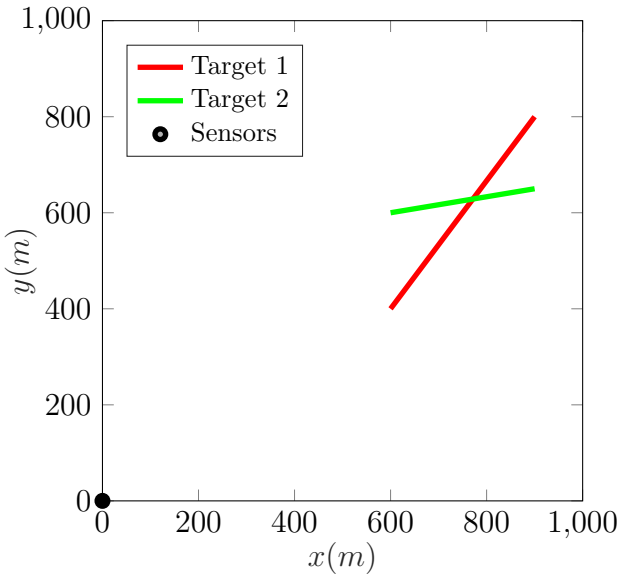


Figure 4.2: Target trajectories for all simulated scenarios and sensor setups. Both sensors are co-located at the origin $(x, y) = (0, 0)$.

4.2 Simulations

In order to test the *single-cluster method* implementation, a challenging simulated scenario that takes place over 100 iterations has been proposed involving crossing targets; a typically difficult problem for MTT algorithms to resolve. Two targets move around inside the Infrared Search and Track (IRST) Field-of-View (FoV) at speeds of approximately 10 knots and 7 knots respectively, with their trajectories crossing one another at approximately $k = 60$. Crossing targets make the data association (DA) problem difficult in tracking scenarios, as sensors with slower update rates such as radar will not generate state estimates fast enough, and track resolution may be diminished. In order to gain a realistic simulation that appears close to the scenario present in the real data set, the sensor properties and update rates used in the simulation are close to those defined in the data sheets for the radar [138].

In the first two experiments, two differing false alarm models will be considered. The first of these will be the typical Poisson model, and secondly, the negative binomial clutter model found in [139], which was introduced earlier in Section 2.3.3. This model will allow for a large variance in the number of false alarms per scan, which will be pertinent in this maritime application, where crests of waves and rough sea conditions are likely to generate larger numbers of false alarms [47, 48, 102]. For these experiments, where the IRST is calibrated on to the radar frame of reference (FoR), an azimuth bias of 3° has been applied to the IRST measurements.

Due to the differing characteristics and sampling rates of the radar and IRST, a third experiment has been developed, where the radar measurements contain the azimuth bias, and the IRST measurements have no bias. This experiment will provide proof that it is still possible to estimate the azimuth bias, even with exploiting the

infrequent radar measurements, rather than the frequent IRST measurements. This also shows that the single-cluster method is flexible, in that it makes little difference which sensor is chosen as the reference sensor.

Both of the offspring processes are implemented using a Gaussian mixture to represent the target states over time. A Gaussian mixture is given by the summation:

$$p\left(\mathbf{x}_{k-1}\right) = \sum_{j=1}^{J_{k-1}} w_{k-1}^{j} \mathcal{N}\left(\mathbf{x}_{k-1}; \mathbf{m}_{k-1}^{j}, \mathbf{P}_{k-1}^{j}\right) \tag{4.4}$$

over a collection of weighted Gaussian distributions with parameters $\left\{w_{k-1}^{j}, \mathbf{m}_{k-1}^{j}, \mathbf{P}_{k-1}^{j}\right\}_{j=1}^{J_{k-1}}$, each with a weight $0 \leq w_{k-1}^{j} \leq 1$. In order to model target births, a measurement-driven birth process is included, as described in [67], and a Hellinger distance [98] merging process is included to merge Gaussian components that sufficiently overlap one another. A standard pruning process is used to remove components with weights lower than τ_{prune}.

The parameters that have been used for these experiments are shown in Table 4.1, and are set to typical values used in PHD filter simulations [91,97]. Although a range of these parameters have been selected based on previous works, it would also be possible to select or estimate some of them in practice. For example, using suitable Signal-to-Noise Ratio (SNR) curves and target models, it should be possible to gain a reasonably accurate probability of detection. Furthermore, if a radar has suitable Constant False Alarm Rate (CFAR) processing built-in, it should be possible to maintain a consistent false alarm rate λ. Varying the parameters in this table will of course affect the overall tracking results that we see in the following output graphs. As some examples:

- having a lower probability of detection will generate fewer measurements on a target, giving us less data to work with, and track accuracy will likely be reduced. Results from Monte-Carlo trials which vary the probability of detection are shown later in this chapter;
- a higher false alarm rate will likely cause more false tracks to be generated, or potentially reduce accuracy in our current tracks if one of the false alarms is used to update a target's state;
- as the measurement noise increases, the accuracy of the sensor measurements themselves will decrease and give poorer output track accuracy. Experiments that vary the measurement noise levels are given in Chapter 5.

The various thresholds listed in the table each play a part in determining when a track should be declared, removed from the track list, and if it should be merged with another track that is close by.

All simulated results have been averaged over 50 Monte Carlo (MC) runs, and the estimated registration angle is taken as the Maximum A Posteriori (MAP) of

Table 4.1: Tracking Parameters - Simulations

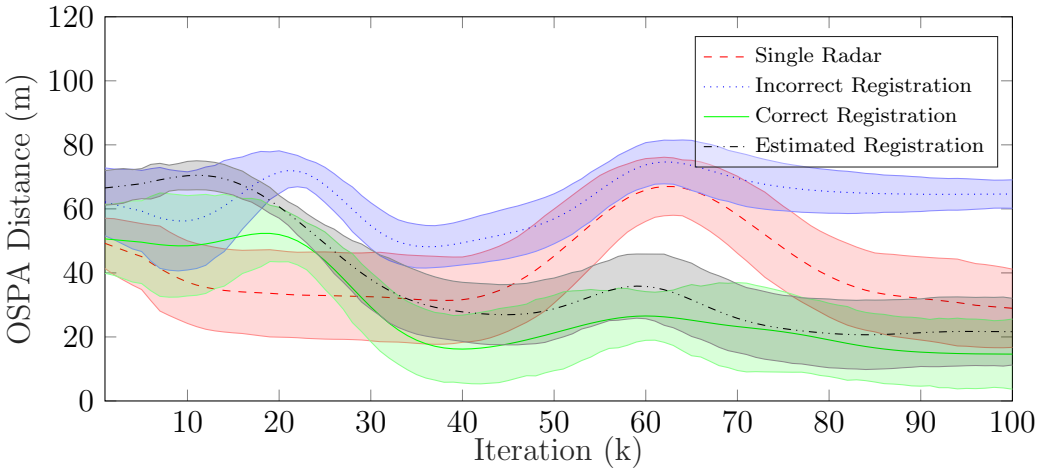
Quantity	Symbol	Sim Value
Detection Probability	p_{d}	$0.7 \rightarrow 0.99$
Survival Probability	p_{s}	0.95
Gating Threshold	τ_{gate}	99%
Pruning Threshold	τ_{prune}	0.001
Merging Threshold	τ_{merge}	0.8
Extraction Threshold	τ_{extract}	0.5
False Alarm Rates	λ_r, λ_c	2, 5
False Alarm Variance	var_r, var_c	10, 50
Birth Intensity	μ_b	1
Acceleration Noise ($m^2 s^{-3}$)	u	1
Radar Measurement Noise (m,°)	σ_{r_r}, σ_{ϕ_r}	5, 0.06
IRST Measurement Noise (°)	σ_{ϕ_c}	0.01

the parent process. The true target trajectories are kept consistent throughout all of the MC runs, however a random set of sensor measurements are generated in each run, based on the measurement models presented in Section 3.4. This will account for the randomness of the measurement noise that would be applied to the measurements in practice. The benchmark result for this work is the case where the radar and IRST system are perfectly registered, giving an optimal result. This is shown as the solid green plot on the following figures.

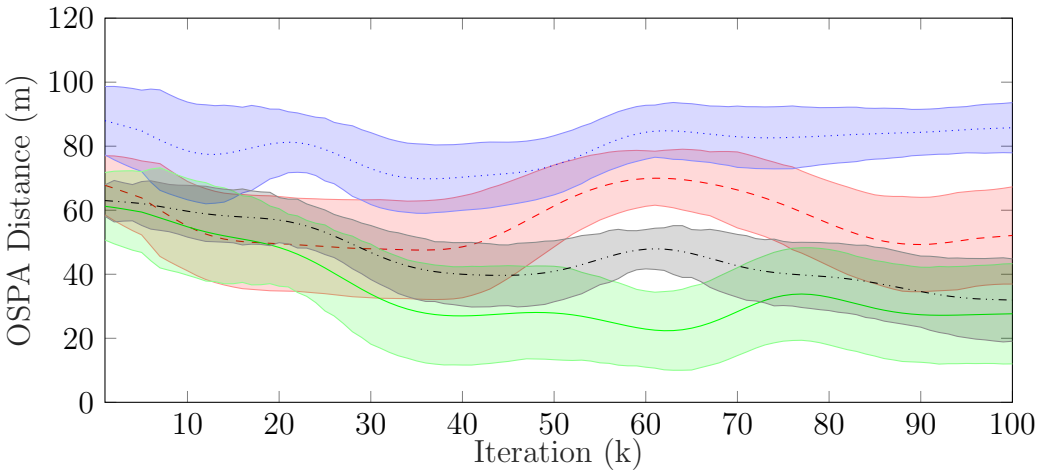
4.2.1 Experiment 1: Poisson Distribution

The first simulation considers the false alarm distribution to be Poisson, where the mean number of false alarms is equal to the variance, giving a consistent number of false alarms per measurement scan. From both Figure 4.3 and Figure 4.4, the importance of taking sensor registration in to account when performing sensor fusion becomes apparent. Firstly in the $p_{\mathrm{d}} = 0.99$ case, by performing tracking with just a single radar (red dashed plot), a single-sensor tracking accuracy benchmark is gained; with an Optimal Subpattern Assignment (OSPA) distance of 29.89 m for the PHD filter, and 26.65 m for the Panjer filter. However, these accuracies can be improved by performing sensor fusion with the IRST measurements and perfect registration (green solid plot). The PHD filter accuracy improves by 15.19 m to 14.70 m, and the Panjer accuracy improves by 14.74 m to 11.91 m.

When an incorrect registration parameter is used and sensor fusion attempted (blue dotted plot), there is a substantial decrease in tracking accuracy, and the OSPA distances increase to values larger than when performing single-sensor tracking. The OSPA distance for the PHD filter increases to 64.55 m and the Panjer

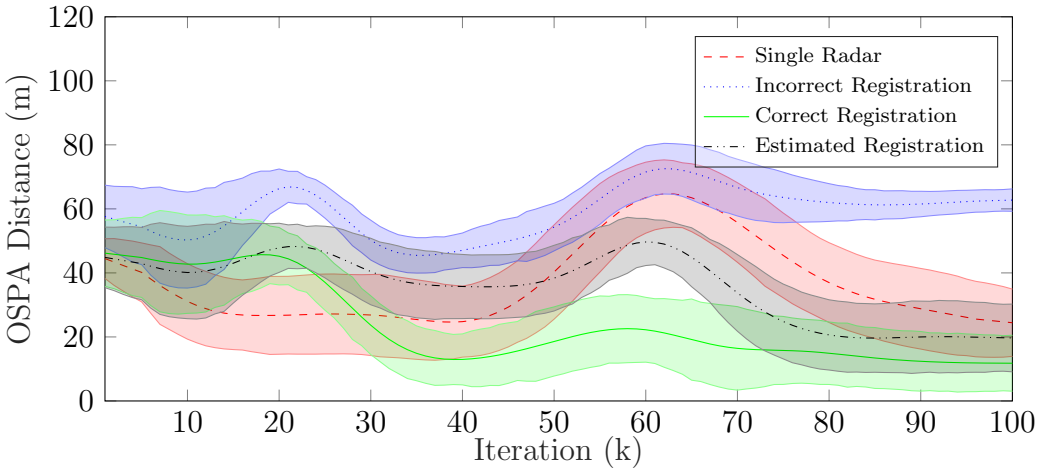


(a) $p_{\mathrm{d}} = 0.99$

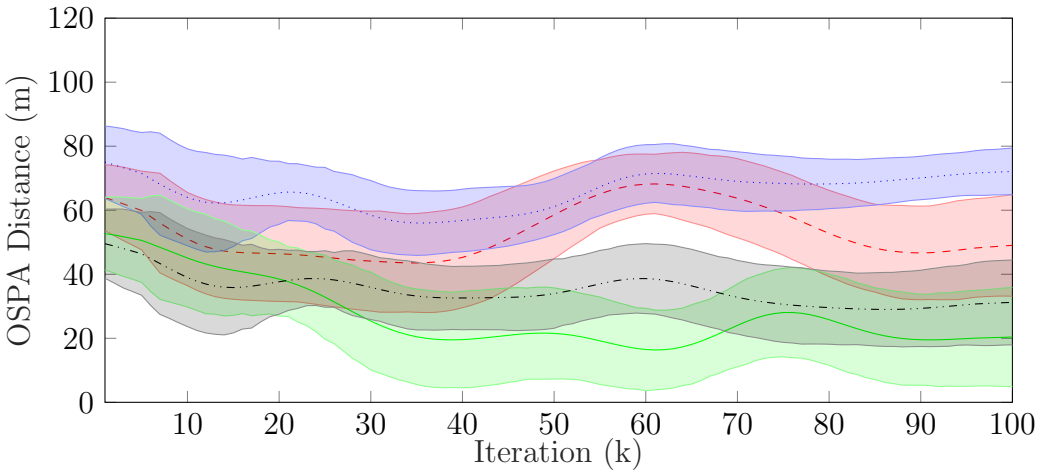


(b) $p_{\mathrm{d}} = 0.85$

Figure 4.3: OSPA Distance over Time, PHD Filter, Poisson Distribution

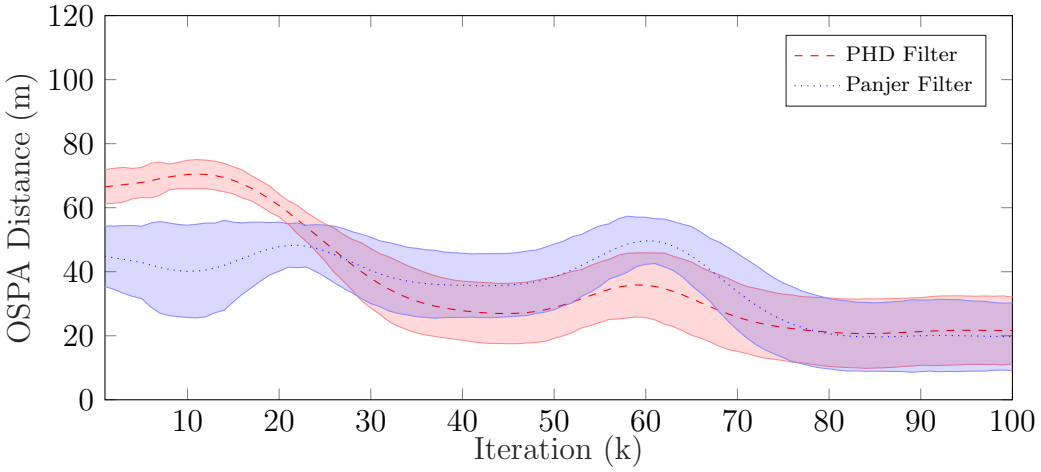


(a) $p_{\mathrm{d}} = 0.99$

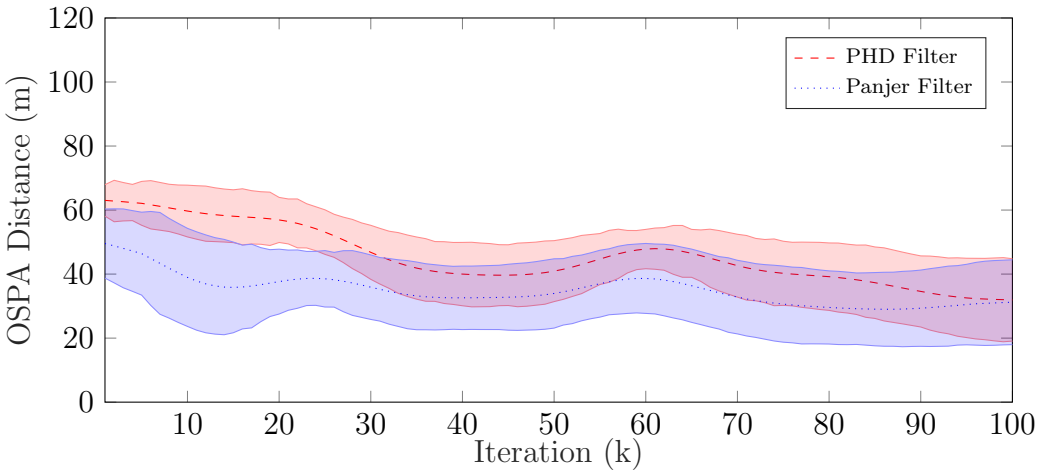


(b) $p_{\mathrm{d}} = 0.85$

Figure 4.4: OSPA Distance over Time, Panjer Filter, Poisson Distribution



(a) $p_{\mathrm{d}} = 0.99$



(b) $p_{\mathrm{d}} = 0.85$

Figure 4.5: Estimated Registration OSPA Distance Comparison, Poisson Distribution

Table 4.2: Average OSPA Distances at $k = 95$, Experiment 1

	p_d	0.85	0.99
Single Radar	PHD	52.12	29.89
	Panjer	47.85	26.65
Incorrect Registration	PHD	85.77	64.55
	Panjer	71.41	62.42
Proposed Method	PHD	32.56	21.71
	Panjer	30.72	20.08
Correct Registration	PHD	27.66	14.70
	Panjer	19.90	11.91

increases to 62.42 m. It can be seen that the proposed solution of using single-cluster methods (black dash-dotted plot) brings the tracking accuracy close to that of the perfect registration case. The PHD filter implementation gives an OSPA distance of 21.71 m, which is 7.01 m from the correct registration result; and the Panjer filter implementation gives an OSPA distance of 20.08 m, 8.17 m from its correct registration result.

The results shown in Figure 4.5 show a direct comparison between the PHD filter implementation of the proposed method, and the Panjer filter implementation of the proposed method. It can be seen that the Panjer filter outperforms the PHD filter in both cases. In the $p_d = 0.99$ result, the $k = 95$ OSPA distances for the PHD and Panjer filters with estimated registration are 21.71 m and 20.08 m respectively, and in the $p_d = 0.85$ case, these are 32.56 m and 30.72 m. This shows that being able to propagate more information about the target population at each time-step can help to improve the tracking accuracy.

The probability of detection also plays a substantial part in this work, and as shown in Figure 4.7a and Figure 4.8a on page 75, as the probability of detection increases, there is an improvement in both the tracking accuracy and the registration parameter estimation. All of the average OSPA distance results for this experiment are given in Table 4.2.

4.2.2 Experiment 2: Negative Binomial Distribution

The second experiment uses a negative binomial clutter distribution. This means that the variance in the number of false alarms is now no longer constrained to be equal to the mean, and allows for the modelling of situations where we could encounter sudden bursts of false alarms. For the maritime application presented in this chapter, large sea swells could generate these bursts of false alarms sporadically [47, 48]. The original PHD filter is based on the assumption that the underlying clutter distribution is Poisson and does not take variance information in to account,

Table 4.3: Average OSPA Distances at $k = 95$, Experiment 2, $p_d = 0.85$

Single Radar	PHD	37.03
	Panjer	32.46
Incorrect Registration	PHD	72.64
	Panjer	68.49
Proposed Method	PHD	36.11
	Panjer	30.95
Correct Registration	PHD	28.06
	Panjer	18.86

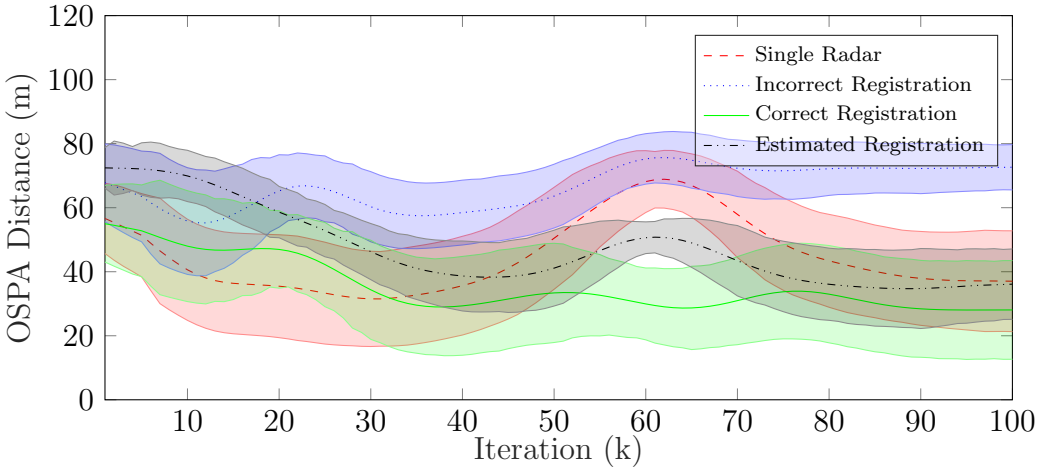
whereas the Panjer filter is more flexible and allows for this to be included in the recursion. The false alarm variance values that are applied during this experiment are given in Table 4.1.

From Figure 4.6, we can see that the Panjer filter can outperform the PHD filter in all cases. At $k = 95$, for the estimated registration case (using the proposed method), the OSPA distance for the PHD filter is 36.11 m, and 30.95 m for the Panjer filter. A much larger performance difference can be seen in Figure 4.8, in that for lower probabilities of detection ($p_d < 0.9$), the performance of the PHD filter is much worse than that of the Panjer filter. The estimates being drawn from the Panjer filter implementation are more consistent across all probabilities of detection simulated.

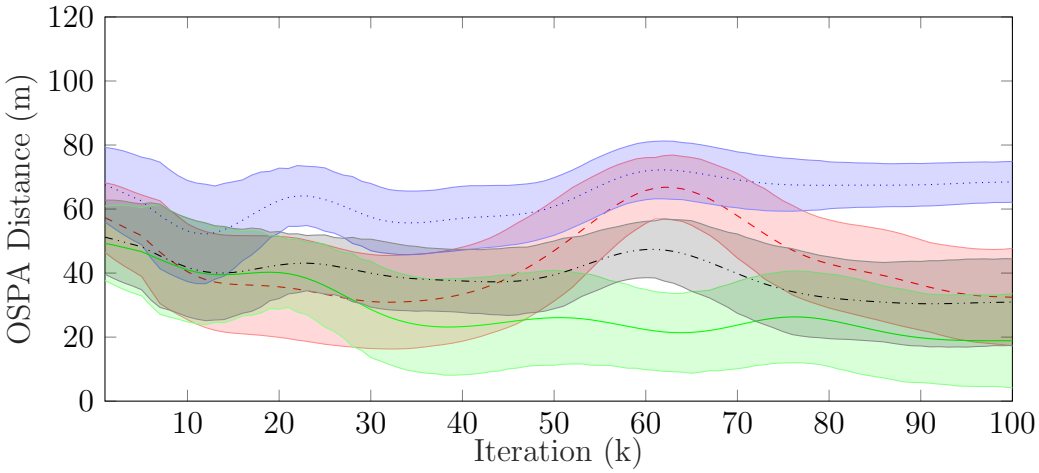
4.2.3 Experiment 3: Alternative Frame of Reference

In the previous experiments, it was assumed that the radar was the reference sensor, and the IRST measurements were subjected to a registration error. The third and final simulated experiment in this chapter concerns the "alternative" registration case, where the radar measurements contain the angular bias and the IRST measurement are unbiased. This experiment was carried out to show it is possible to use either sensor as the reference frame to perform fusion in.

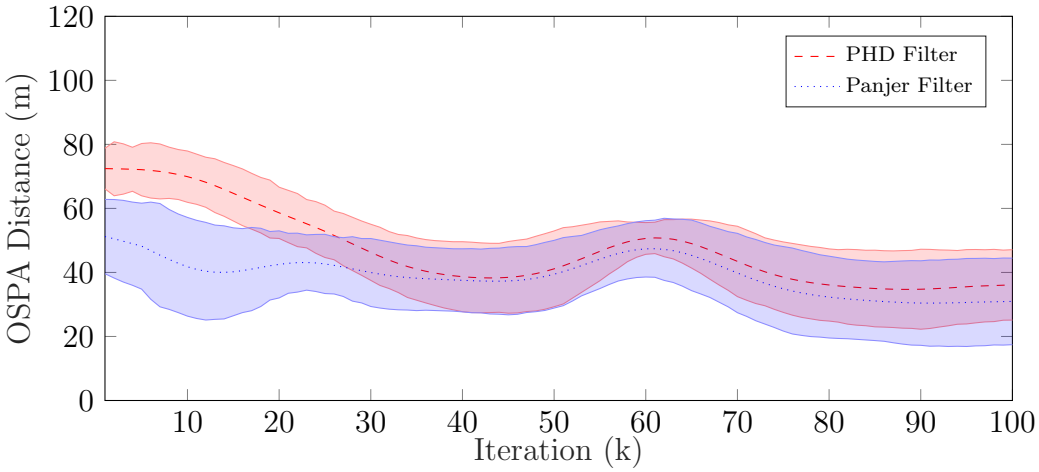
From Figure 4.9a on page 77, as expected, the tracking accuracy when using a single radar has decreased, and the result for the uncalibrated set of sensors is still very poor. The result using the proposed estimation method is close to the correctly registered result, with an OSPA distance of 27.53 m at $k = 95$. The results shown in Figure 4.9a are tabulated in Table 4.4. As the probability of detection p_d increases, we see similar behaviour to the previous experiments; both the target tracking accuracy and the registration estimation accuracy improve.



(a) PHD Filter, $p_{\mathrm{d}} = 0.85$

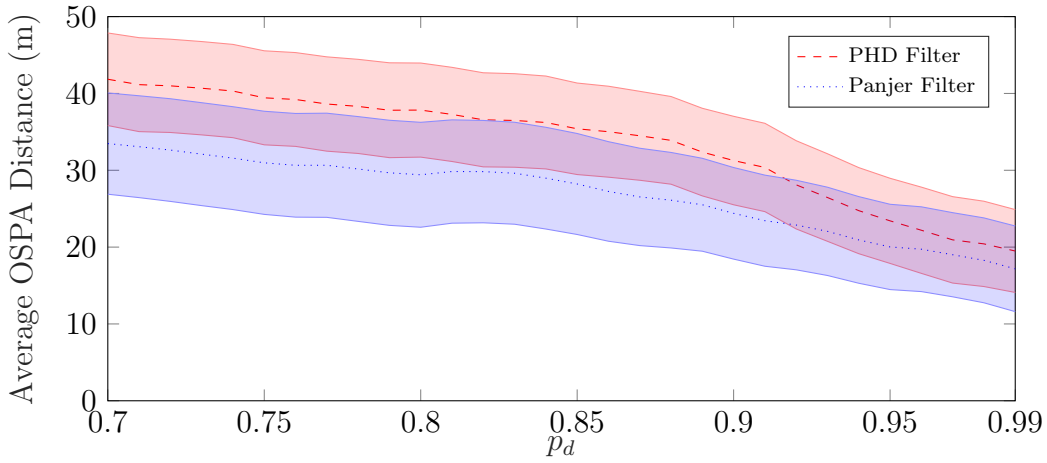


(b) Panjer Filter, $p_{\mathrm{d}} = 0.85$

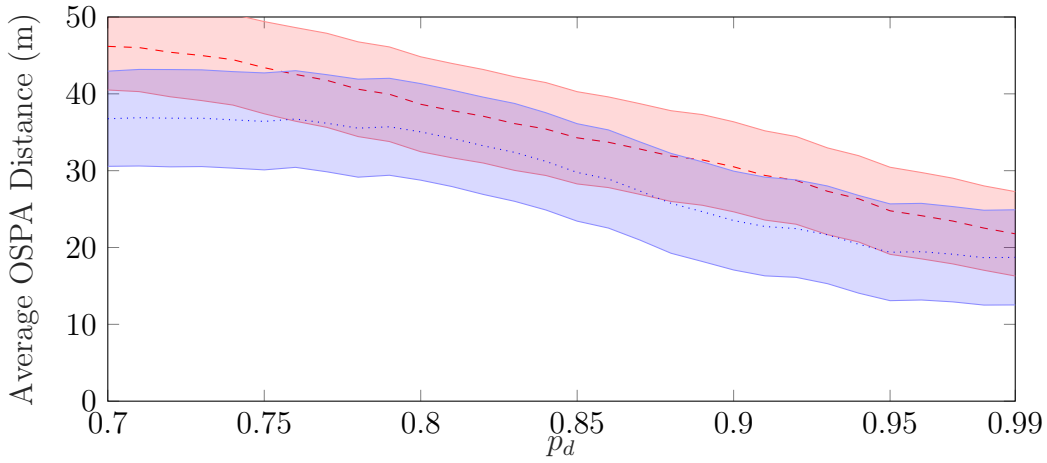


(c) OSPA comparison

Figure 4.6: OSPA Distance over Time, Negative Binomial Distribution



(a) Poisson Distributed

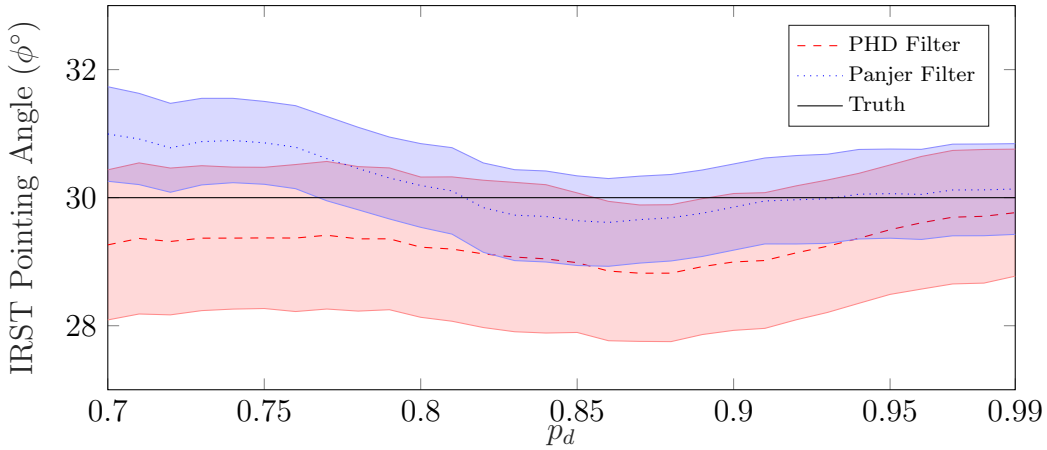


(b) Negative Binomial Distributed

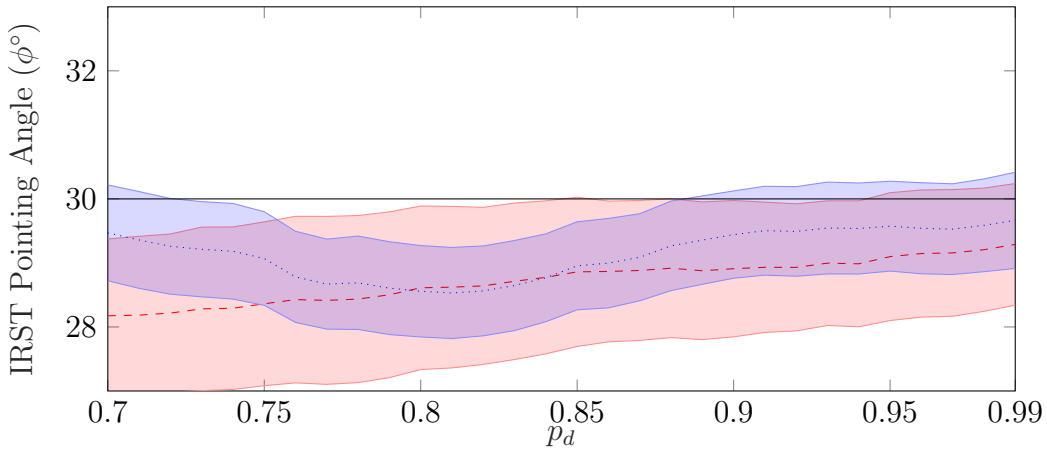
Figure 4.7: OSPA Distance over p_{d}. The average OSPA Distance is taken from between Iteration 80 and Iteration 100.

Table 4.4: Average OSPA Distances at $k = 95$, Experiment 3, $p_{\mathrm{d}} = 0.99$

Single Radar	68.90
Incorrect Registration	84.04
Proposed Method	27.53
Correct Registration	17.10

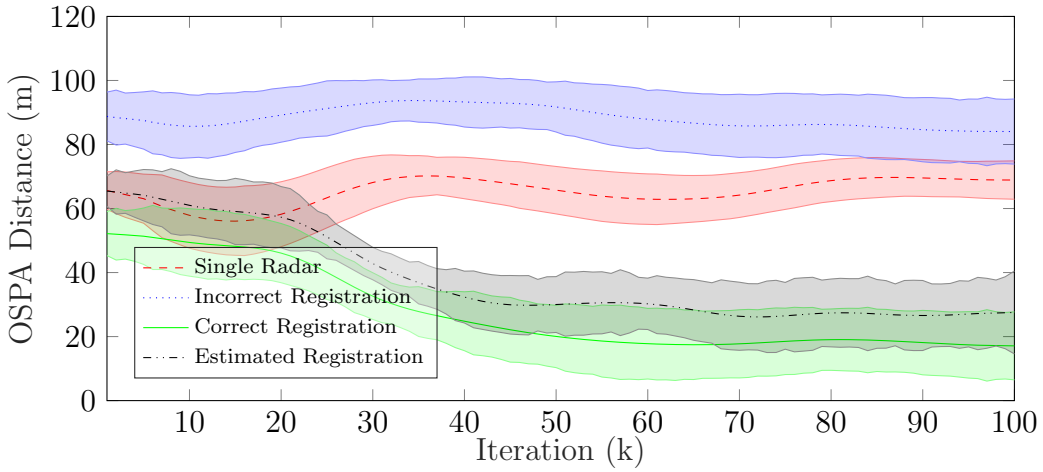


(a) Poisson Distributed

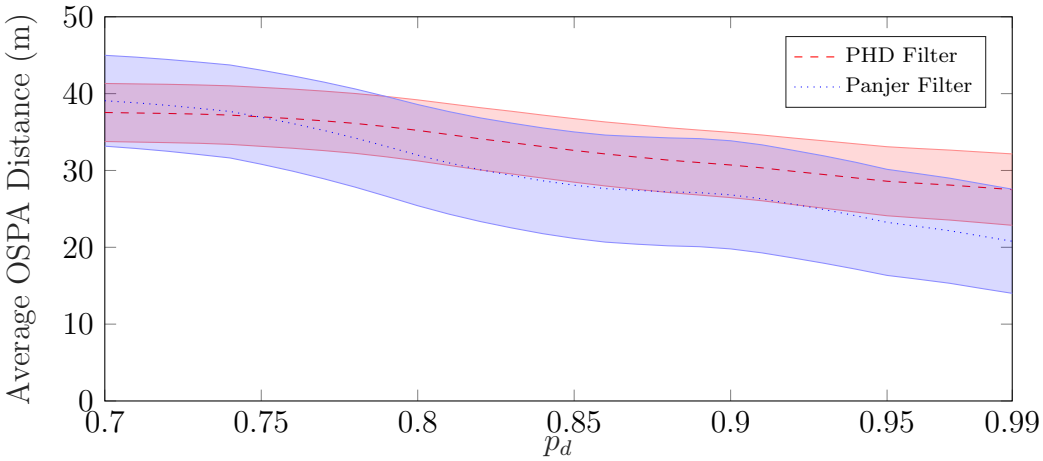


(b) Negative Binomial Distributed

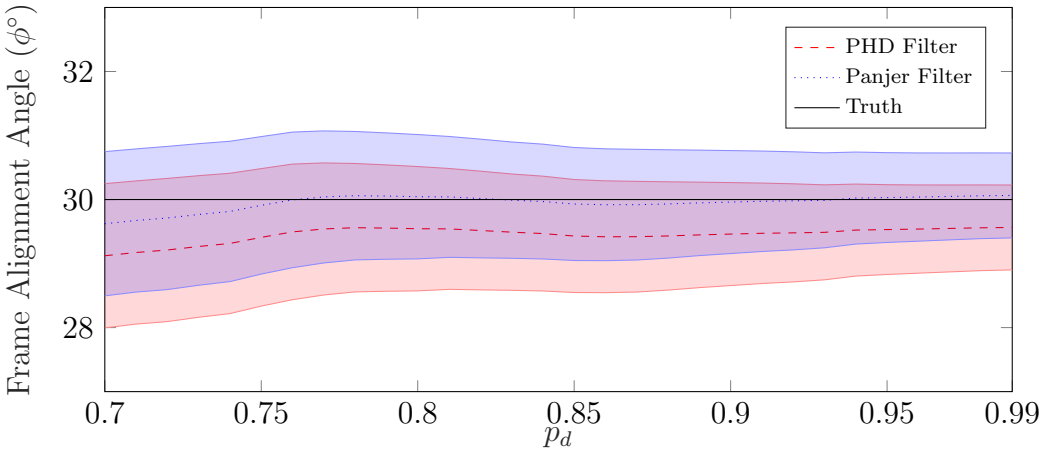
Figure 4.8: IRST Pointing Angle over p_{d}. The average pointing angle is taken from between Iteration 80 and Iteration 100.



(a) Radar bias, $p_{\mathrm{d}} = 0.99$, PHD filter



(b) Average OSPA Distance



(c) Estimated Alignment Angle w.r.t. IRST frame

Figure 4.9: Results for alternative FoR scenario.

Figure 4.10: Sensor setup used for recording real data set with radar system in the background and IRST in the foreground. © Crown copyright, 2019.

4.3 Real Data - Collaboration with Dstl

4.3.1 Discussion

As an extension to this work, the proposed algorithm was tested on a portion of real data, made available by Dstl. The sensors shown in Figure 4.10 were located inside a shipping container next to the sea wall at Fort Blockhouse, Gosport, Portsmouth Harbour, U.K.. The radar in use was a Kelvin Hughes SharpEye system [138], which was operating in a 360° scanning mode, giving a large surveillance region, but with a slow update rate of approximately 2.5 s or 0.4 Hz. The radar was equipped with a number of signal processing and detection techniques, allowing for a range of different types of maritime target to the detected effectively. The low-profile antenna gives an azimuth beam-width of less than 1° at the 3 dB point [138]. Advanced techniques such as monopulse [36] to further improve the radar's accuracy in azimuth were not in use; target localisation inside the radar beam was not possible.

The IRST system was a recently-developed sensor for research purposes with little information published in the open literature. IRST is an effective method for detecting targets that emit infrared signatures. The main difference between this and the radar system is the wavelength that they operate at. The wavelength of the IRST is much shorter than that of the radar which, in turn, gives much better angular resolution. There is however, one main drawback of using the IRST; its detection performance can be largely affected by atmospheric and weather conditions. The aim is to exploit the better angular accuracy, and the higher update rate of the IRST, to gain more accurate tracking information through the use of sensor fusion.

With the sensors looking out over the Solent, there were a large number of

Table 4.5: Tracking Parameters - Real Data

Quantity	Symbol	Value
Detection Probability	p_{d}	0.6
Survival Probability	p_{s}	0.95
Gating Threshold	τ_{gate}	99%
Pruning Threshold	τ_{prune}	0.001
Merging Threshold	τ_{merge}	0.6
Extraction Threshold	τ_{extract}	0.5
False Alarm Rates	λ_r, λ_c	5, 10
Birth Intensity	μ_b	0.01
Acceleration Noise ($m^2 s^{-3}$)	u	3
Radar Measurement Noise (m,°)	σ_{r_r}, σ_{ϕ_r}	3.873, 0.059
IRST Measurement Noise (°)	σ_{ϕ_c}	0.016

opportunities for detecting maritime targets. A number of instrumented targets were present in the scene, as well as a large amount of background traffic, including ferries and cargo ships passing through, allowing for more targets to be tracked, and therefore increase the MOL and improve the calibration accuracy. In the segment of data used in this piece of work, a large and persistent target crosses the FoV approximately 2 km away from the sensors' location. The target is clearly visible for 50 s and then disappears from the IRST FoV.

4.3.2 Results

It can be seen in Figure 4.11 that after the initial estimate of $-10°$, and the ensuing transition period, an angular registration error of between 1.5° and 2° is estimated in the parent process. After the short transition period, the estimated angle only deviates a small amount which may be due to external factors, such as platform vibrations, or due to weather conditions such as high winds for example. It is also possible that the omission of elevation data from the problem may account for this small deviation. This systematic error, or bias, could account for human error when the sensors were attached to the platform, as installation is often done by eye.

The main attraction of performing sensor fusion is apparent in Figure 4.12 and Figure 4.13. The track shown in Figure 4.12 appears smooth, and could represent a realistic maritime target, moving with a constant velocity. There are quite large gaps between consecutive radar measurement updates however. These gaps can however be filled in by using IRST measurements as shown in Figure 4.13. With range measurements only available as a part of the radar measurements, there are two larger jumps in the track at $(-2000, 400)$ and $(-1850, 450)$. As shown on the figure, these are points where range information has become available again and

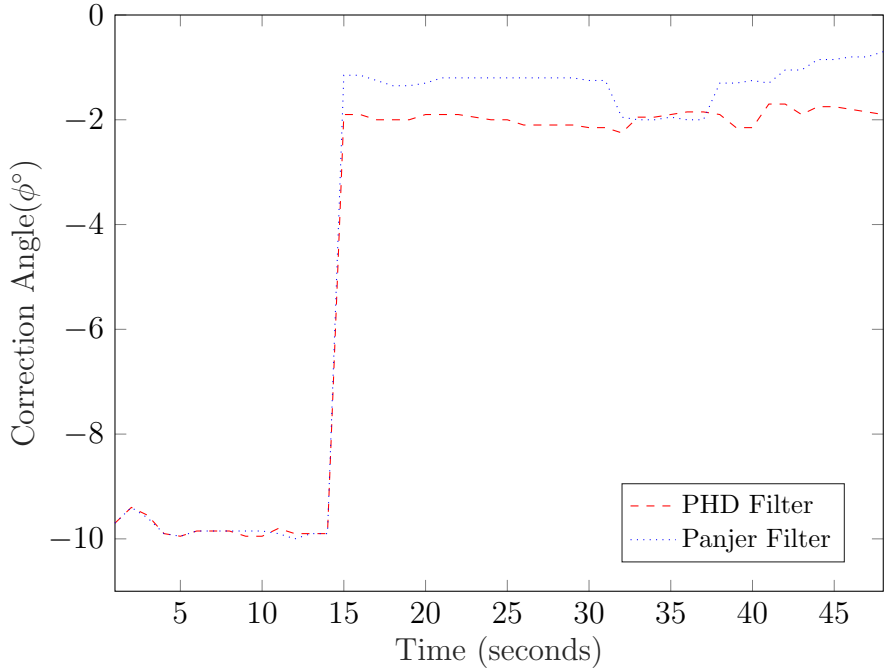


Figure 4.11: Estimated correction angles for real data scenario.

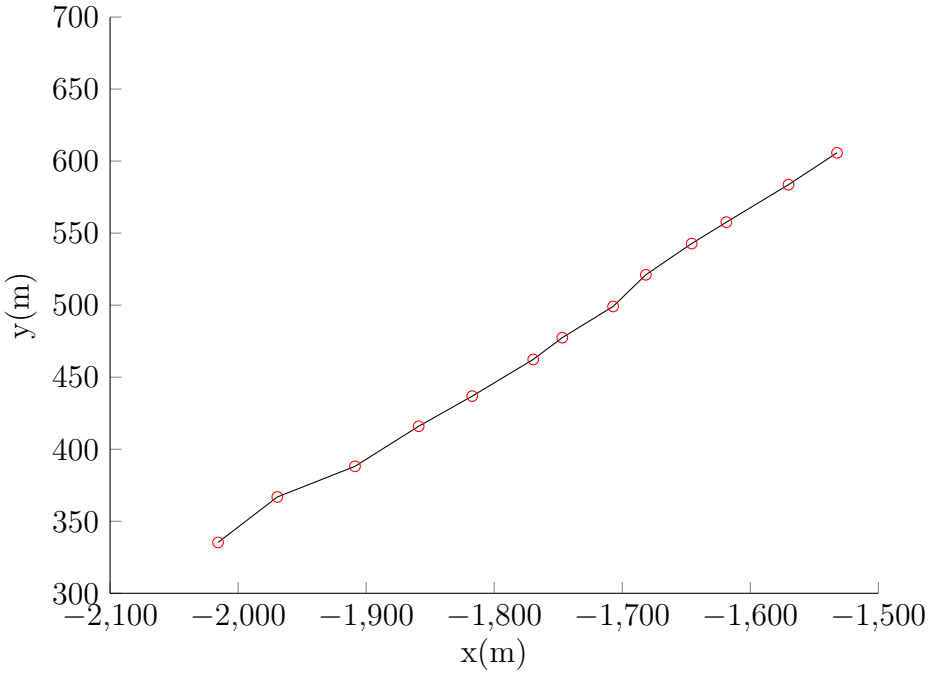


Figure 4.12: Real target track using only radar measurements.

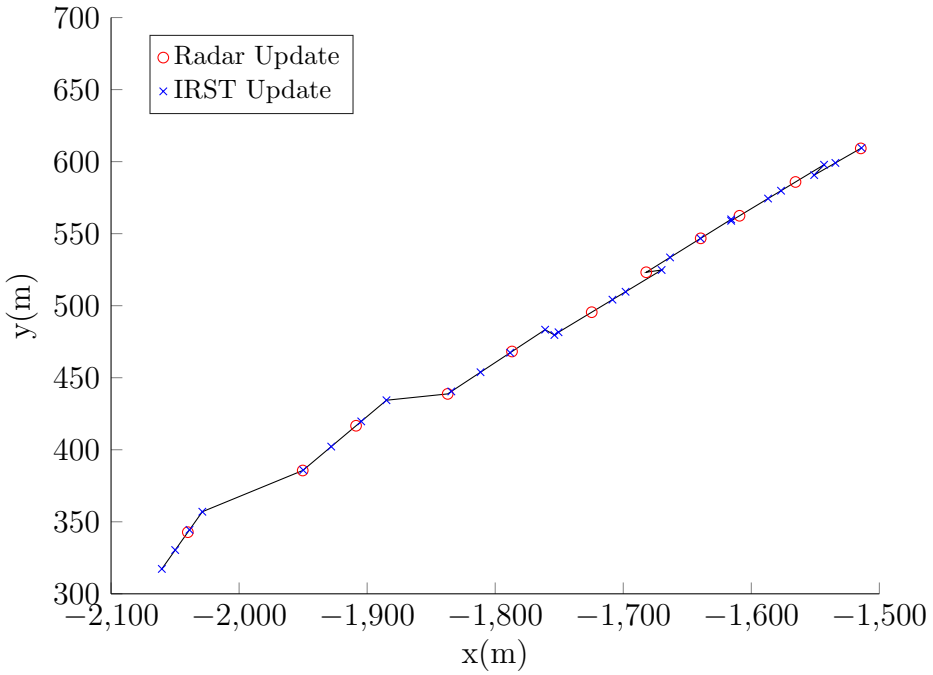


Figure 4.13: Real target track using radar and IRST measurements.

updated appropriately.

4.4 Conclusions

In conclusion of this work, it has been demonstrated that *single-cluster methods* could provide a suitable solution for performing joint sensor registration and fusion for asynchronous and heterogeneous sensors. The simulation results on the crossing targets scenario provide convincing evidence to show the importance of taking sensor registration in to account before attempting any form of data fusion. The application of this technique to a set of real data has also been successful, and a plausible offset in the sensors pointing angle was found.

It was necessary to only use a subset of the available measurements, which gave more weight to the sparse range-bearing measurements from the radar. Fundamentally, by only using the slow update rate available from the radar, there are long periods where no target track estimates are given, and therefore track resolution is lost.

Chapter 5

Vector-Type Methods for Improved Accuracy and Efficiency

The solution that was proposed in Chapter 4 for resolving the joint registration and fusion problem was robust and provided reasonable estimation accuracy. However, by using vector-type tracking methods, rather than point process methods, further advantages can be gained such as the ability to maintain target identities over time, and therefore have a unique track history for each individual target. Vector-type methods resolve the data association problem during the tracking process, and can distinguish individual targets within a population, whereas point process methods such as the Probability Hypothesis Density (PHD) filter do not attempt to associate measurements to targets explicitly, and individual targets are not uniquely distinguishable [80].

The grid-based method used for estimating the sensor registration parameter was restrictive; it may be possible in extreme cases that the true value of the parameter could exceed the bounds of the grid. This problem will also be addressed in this chapter, as the grid-based method is removed, and replaced with a particle filter and resampling approach, which should improve the flexibility and efficiency of the algorithm. By improving the accuracy and efficiency in the two-sensor case, this should lead to larger gains when looking at the many-sensor case in Chapter 6. The work carried out in this chapter has been published, in part, in publications C2 [140] and C3 [103], as listed in Chapter 1.

The main contributions of this chapter, and advancements from Chapter 4, include:

- the extension of a Multi-Target Tracking (MTT) routine based on Message Passing (MP) that allows for simultaneous sensor registration and fusion; the proposed technique also addresses the data association (DA) problem efficiently using the Sum-Product Algorithm for Data Association (SPADA) [77];
- the derivation of a suitable Multi-Object Likelihood (MOL) for the particle

Belief Propagation (BP) implementation presented, so that the algorithm can be placed in to the hierarchical framework;

- the removal of the grid-based method, which is replaced with a Sequential Monte-Carlo (SMC) approach, where the particle distribution representing the registration parameter(s) is sampled and resampled at each time-step; this gives more flexibility and will reduce the computational complexity when more parameters need to be estimated (Chapter 6);

- the introduction of the multi-radar use case, which follows on from the work presented in [140]; a full comparison with the point process approach presented in Chapter 4 is given as a benchmark.

5.1 Message Passing for MTT

The use of Belief Propagation (BP), also known as Message Passing (MP), or the Sum-Product Algorithm (SPA), is widespread in the signal processing literature, and has been applied to a wide range of problems such as channel coding/decoding [141], cooperative sensor localisation [142], and data association [143]. The use of MP for MTT brings in a number of distinct advantages:

- "soft" association probabilities can be calculated efficiently;

- a principled technique for deriving MTT methods within a Bayesian inference framework;

- the MP methods generalise previous MTT techniques such as Joint Probabilistic Data Association (JPDA) [82, 83];

- introduces scalable methods for MTT that are suitable for scenarios that involve large numbers of targets and sensors. This point in particular will be investigated in Chapter 6, when the proposed registration and fusion algorithms are applied to larger sensor networks.

In contrast to the point process-based method previously used in the offspring process, which gave an approximation of the joint posterior probability density function (pdf) that represented the target distribution, the use of MP will give the marginal posterior pdfs for each of the individual targets. These pdfs can then be used can then be used to perform Bayesian estimation of the target states. The most recent methods of BP for MTT [22, 23], which will be extended here, scales quadratically in the number of targets, linearly in the number of sensors, and linearly in the number of measurements per sensor. The work by Meyer et al. in [22] showed that MP-based algorithms should perform more efficiently than point process methods.

The following sections will closely follow the notation found in [22, 103, 140]. Let us start by defining some notation that will be used throughout this chapter:

$\mathbf{x}_{k,n}$: target state vector *for target* n at time k;

$r_{k,n} \in \{0, 1\}$: target existence indicator for target n at time k;

$\mathbf{y}_{k,n} = \{\mathbf{x}_{k,n}, r_{k,n}\}$: augmented state for target n at time k;

$\mathbf{a}_k$: target-oriented association variables at time k;

$\mathbf{b}_k$: measurement-oriented association variables at time k;

$s_\star$: spatial distribution of false alarms;

$\Psi(a_{k,n}, b_{k,m})$: exclusion enforcing function for association assumptions;

$\tilde{f}(\mathbf{x}_{k,n}, r_{k,n})$: belief approximation of target n at time k

5.1.1 Algorithm Formulation

By running an iterative BP scheme on the factor graph, approximations of the marginal posterior pdfs can be found in a very efficient manner. Because the factor graph contains loops, a specific order for passing messages should be defined, based on the following rules:

1. Messages should only flow forwards in time [142].

2. Iterative message passing is only performed for the DA function.

With these, the algorithm can be summarised with the following BP operations. Firstly, the state prediction step is performed for all targets such that:

$$\alpha(\mathbf{x}_{k,n}, r_{k,n}) = \sum_{r_{k-1,n}} \int f(\mathbf{x}_{k,n}, r_{k,n} | \mathbf{x}_{k-1,n}, r_{k-1,n}) \times \tilde{f}(\mathbf{x}_{k-1,n}, r_{k-1,n}) \mathrm{d}\mathbf{x}_{k-1,n} \quad (5.1)$$

where $\tilde{f}(\mathbf{x}_{k-1,n}, r_{k-1,n})$ contain the target beliefs from the previous time-step $k-1$. This follows the same form as the Chapman-Kolmogorov equation, where the underlying dynamic model is applied to the previous target beliefs to gain the predicted beliefs for the current time k.

After the prediction step, correlation and association steps are used to help form unique tracks on individual targets. The correlation step can be expressed as

$$\beta(a_{k,n}) = \sum_{r_{k,n} \in \{0,1\}} \int \upsilon(\mathbf{x}_{k,n}, r_{k,n}, a_{k,n}; \mathbf{Z}_k) \times \alpha(\mathbf{x}_{k,n}, r_{k,n}) \mathrm{d}\mathbf{x}_{k,n} \quad (5.2)$$

where

$$v(\mathbf{x}_{k,n}, r_{k,n}=1, a_{k,n}|\mathbf{Z}_k) = \begin{cases} \frac{f(\mathbf{z}_{k,m}|\mathbf{x}_{k,n})}{s_\star(\mathbf{z}_{k,m})}\frac{p_\mathrm{d}}{\lambda} & a_{k,n}=m\in M_k \\ 1-p_\mathrm{d} & a_{k,n}=0 \end{cases}, \tag{5.3}$$

$$v(\mathbf{x}_{k,n}, r_{k,n}=0, a_{k,n}|\mathbf{Z}_k) = 1(a_{k,n}), \tag{5.4}$$

λ represents the average false alarm rate as per the Poisson point process and m is the measurement index, $m \in \{1,\ldots,M_k\}$. As with the Poisson point process used in the previous chapter, λ will define the cardinality (number) of false alarm measurements, and these false alarms will be randomly generated within the limits of the spatial distribution $s_\star$. The data association (DA) algorithm SPADA [77] used here is also based on BP, and has been shown to be more efficient than other classical techniques. The algorithm is initialised at iteration $p=0$ using

$$\zeta^{(0)}_{n\to m}(b_{k,m}) = \sum_{a_{k,n}=0}^{M_k} \beta(a_{k,n})\Psi(a_{k,n}, b_{k,m}). \tag{5.5}$$

Once initialised, the two following equations are executed in a loop from $p=1$ until a final iteration P is reached:

$$\upsilon^{(p)}_{m\to n}(a_{k,n}) = \sum_{b_{k,m}=0}^{N} \Psi(a_{k,n}, b_{k,m}) \times \prod_{n'\in N\setminus\{n\}} \zeta^{(p-1)}_{n'\to m}(b_{k,m}) \tag{5.6}$$

$$\zeta^{(p)}_{n\to m}(b_{k,m}) = \sum_{a_{k,n}=0}^{M_k} \beta(a_{k,n})\Psi(a_{k,n}, b_{k,m}) \times \prod_{m'\in M_k\setminus\{m\}} \upsilon^{(p)}_{m'\to n}(a_{k,n}). \tag{5.7}$$

For the simulations presented later in Section 5.4, the association scheme will run for a pre-defined number of iterations $P=30$. After the final loop is completed, the message to be passed out to the update step is given by

$$\eta(a_{k,n}) = \prod_{m=1}^{M_k} \upsilon^{(P)}_{m\to n}(a_{k,n}). \tag{5.8}$$

Depending on whether a target still exists or not, the update equation will either be

$$\gamma(\mathbf{x}_{k,n}, r_{k,n}=1) = \sum_{a_{k,n}=0}^{M_k} v(\mathbf{x}_{k,n}, r_{k,n}=1, a_{k,n}; \mathbf{Z}_k)\eta(a_{k,n}) \tag{5.9}$$

$$\gamma(\mathbf{x}_{k,n}, r_{k,n}=0) = \eta(a_{k,n}=0). \tag{5.10}$$

Lastly, the beliefs that represent the marginal posterior pdfs are obtained by

$$\tilde{f}(\mathbf{x}_{k,n}, r_{k,n} = 1) = \frac{1}{C_{k,n}} \alpha(\mathbf{x}_{k,n}, r_{k,n} = 1)\gamma(\mathbf{x}_{k,n}, r_{k,n} = 1) \tag{5.11}$$

$$\tilde{f}(\mathbf{x}_{k,n}, r_{k,n} = 0) = \frac{1}{C_{k,n}} \alpha(\mathbf{x}_{k,n}, r_{k,n} = 0)\gamma(\mathbf{x}_{k,n}, r_{k,n} = 0) \tag{5.12}$$

where $C_{k,n}$ contains normalisation constants -

$$\int \alpha(\mathbf{x}_{k,n}, r_{k,n} = 1)\gamma(\mathbf{x}_{k,n}, r_{k,n} = 1)\mathrm{d}\mathbf{x}_{k,n} + \alpha(\mathbf{x}_{k,n}, r_{k,n} = 0)\gamma(\mathbf{x}_{k,n}, r_{k,n} = 0). \tag{5.13}$$

5.2 Multi-Object Likelihood Derivation

5.2.1 Derivation of Posterior

In order to use this type of BP algorithm for sensor registration in the single-cluster framework, a suitable MOL is required to link the processes together, as described in Section 2.3.3. The equations presented in this subsection can be found in [22]. First, it should be assumed that the prior distribution from $k-1$ can be factorised as

$$p(\mathbf{X}_{k-1}, \mathbf{r}_{k-1}|\mathbf{Z}_{k-1}) = \prod_{n=1}^{N} p(\mathbf{x}_{k-1,n}, r_{k-1,n}|\mathbf{Z}_{k-1}), \tag{5.14}$$

and the state evolution can also be factorised as

$$f(\mathbf{X}_k, \mathbf{r}_k|\mathbf{X}_{k-1}, \mathbf{r}_{k-1}) = \prod_{n=1}^{N} f(\mathbf{x}_{k,n}, r_{k,n}|\mathbf{x}_{k-1,n}, r_{k-1,n}), \tag{5.15}$$

such that the joint target density can be formed as a product of all of the individual target densities. The prediction equation for $\{\mathbf{x}_{k,n}, r_{k,n}\}_{n=1}^{N}$ can therefore be written as

$$p(\mathbf{X}_k, \mathbf{r}_k|\mathbf{Z}_{k-1}) = \int f(\mathbf{X}_k, \mathbf{r}_k|\mathbf{X}_{k-1}, \mathbf{r}_{k-1})p(\mathbf{X}_{k-1}, \mathbf{r}_{k-1}|\mathbf{Z}_{k-1})\mathrm{d}\mathbf{X}_{k-1}. \tag{5.16}$$

which is in the same form as the Chapman-Kolmogorov equation. For simplicity, this prediction equation will be written as a product of marginal predictions, such that

$$p(\mathbf{X}_k, \mathbf{r}_k|\mathbf{Z}_{k-1}) = \prod_{n=1}^{N} \alpha(\mathbf{x}_{k,n}, r_{k,n}) \tag{5.17}$$

where $\alpha(\mathbf{x}_{k,n}, r_{k,n})$ is a marginal prediction in the same form as equation (5.1):

$$\alpha(\mathbf{x}_{k,n}, r_{k,n}) = \sum_{r_{k,n}} \int f(\mathbf{x}_{k,n}, r_{k,n}|\mathbf{x}_{k-1,n}, r_{k-1,n})p(\mathbf{x}_{k-1,n}, r_{k-1,n}|\mathbf{z}_{k-1})\mathrm{d}\mathbf{x}_{k-1,n}. \tag{5.18}$$

The next step is to introduce the target-oriented association variables a_k into the formulation. This gives an update equation, based on Bayes' rule, for the MP implementation:

$$p(\mathbf{X}_k, \mathbf{r}_k, \mathbf{a}_k|\mathbf{Z}_k, \mathbf{Z}_{1:k-1}) = p(\mathbf{Z}_k|\mathbf{X}_k, \mathbf{r}_k, \mathbf{a}_k) \times \frac{p(\mathbf{a}_k|\mathbf{X}_k, \mathbf{r}_k, \mathbf{Z}_{1:k-1})p(\mathbf{X}_k, \mathbf{r}_k|\mathbf{Z}_{1:k-1})}{p(\mathbf{Z}_k|\mathbf{Z}_{1:k-1})} \tag{5.19}$$

where $\mathbf{a}_k$ are the target-oriented association variables, $p(\mathbf{Z}_k|\mathbf{X}_k, \mathbf{r}_k, \mathbf{a}_k)$ is the single-object association likelihood, $p(\mathbf{X}_k, \mathbf{r}_k|\mathbf{Z}_{1:k-1})$ contains the predicted target states obtained from (5.16), and $p(\mathbf{Z}_k|\mathbf{Z}_{1:k-1})$ is the evidence term, which is necessary for the derivation of the MOL function.

This then leads to an application of the stretching process discussed in Section 2.4. In order to stretch the graph in this case, a latent random variable $\mathbf{b}_k$ is introduced, which represents the measurement-oriented association variables. This information stored in $\mathbf{b}_k$ can be directly derived from $\mathbf{a}_k$, giving a seemingly redundant formulation. However this is a critical step in finding a more detailed factor graph, and reducing the dimensionality. This stretching process may introduce loops into a part of the factor graph. Having loops means that the messages or beliefs being passed in the loopy part of the graph are no longer exact, and are only *approximations.* By exploiting the final factorisation given in [22, Eq. (27)], and also exploiting both $\mathbf{a}_k$ and $\mathbf{b}_k$ with the update equation given in (5.19), it can be shown that

$$p(\mathbf{X}_k, \mathbf{r}_k, \hat{\mathbf{a}}_k|\mathbf{Z}_k) \propto \Psi(\hat{\mathbf{a}}_k)\frac{\prod_{n=1}^{N} v(\mathbf{x}_{k,n}, r_{k,n}, a_{k,n}|\mathbf{Z}_k) \prod_{n=1}^{N} \alpha(\mathbf{x}_{k,n}, r_{k,n})}{p(\mathbf{Z}_k|\mathbf{Z}_{1:k-1})} \tag{5.20}$$

where $v(\mathbf{x}_{k,n}, r_{k,n}, a_{k,n}|\mathbf{Z}_k)$ are computed through equations (5.3) and (5.4), $\hat{\mathbf{a}}_k = \{\mathbf{a}_k, \mathbf{b}_k\}$, and $\Psi(\hat{\mathbf{a}}_k)$ is an "indicator" term that excludes all infeasible association events [144].

5.2.2 MOL for MP Implementation

In order to gain the posterior in equation (5.20) from [22], there was no conditioning on the parameters to be estimated $\mathbf{q}$; this was never taken into account in the article, and was assumed known. Equation (5.20) can be re-written with this conditioning:

$$p(\mathbf{X}_k, \mathbf{r}_k, \hat{\mathbf{a}}_k|\mathbf{Z}_k, \mathbf{q}_k) \propto \Psi(\hat{\mathbf{a}}_k)\frac{\prod_{n=1}^{N} v(\mathbf{x}_{k,n}, r_{k,n}, a_{k,n}|\mathbf{Z}_k, \mathbf{q}_k) \prod_{n=1}^{N} \alpha(\mathbf{x}_{k,n}, r_{k,n}|\mathbf{q}_k)}{p(\mathbf{Z}_k|\mathbf{Z}_{1:k-1}, \mathbf{q}_k)} \tag{5.21}$$

The next step is to rearrange equation (5.20). The evidence term can be moved from the denominator of the right-hand side to the left-hand side, and by marginalising

over $\mathbf{a}_k$, $\mathbf{X}_k$ and $\mathbf{r}_k$, this therefore gives an expression for the evidence term:

$$p(\mathbf{Z}_k|\mathbf{Z}_{1:k-1},\mathbf{q}_k) \propto \sum_{\mathbf{a}_k}\sum_{r_k}\int\cdots\int\prod_{n=1}^{N} v(\mathbf{x}_{k,n},r_{k,n},a_{k,n}|\mathbf{Z}_k,\mathbf{q}_k) \times \alpha(\mathbf{x}_{k,n},r_{k,n}|\mathbf{q}_k)\mathrm{d}\mathbf{x}_{k,1},\ldots,\mathrm{d}\mathbf{x}_{k,n}. \tag{5.22}$$

The product over N is removed from the multidimensional integral, so that this expression can be simplified further using [22, Eq. (31)]:

$$\beta(a_{k,n}) = \sum_{r_{k,n}}\int v(\mathbf{x}_{k,n},r_{k,n},a_{k,n}|\mathbf{Z}_k,\mathbf{q}_k)\alpha(\mathbf{x}_{k,n},r_{k,n}|\mathbf{q}_k)\mathrm{d}\mathbf{x}_{k,n}, \tag{5.23}$$

so that the final expression for the MOL of the MP implementation is:

$$p(\mathbf{Z}_k|\mathbf{Z}_{1:k-1},\mathbf{q}_k) \propto \sum_{\mathbf{a}_k}\prod_{n=1}^{N}\beta(a_{k,n}) = \hat{\ell}_k(\mathbf{q}_k|\mathbf{Z}_k) \tag{5.24}$$

The $\beta(a_{k,n})$ terms can be interpreted as an approximation of the single-target association weights commonly found in the Probabilistic Data Association (PDA) and JPDA filters [21]. This MOL is based on the correlation and association information, which from a wider perspective, follows the conclusions proposed by Vermaak, Maskell and Briers in [34], and by Li et al. in [14], in that the performance of the association process could have an inherent effect on the sensor registration estimation.

The final equation given in (5.24) can be directly implemented between the correlation and association steps of the offspring process, on the matrix of size $((M_k+1)\times N)$ containing the weights, where M_k is the number of current measurements. An extra row is required in this matrix to take account of the possibility that the measurement will not correlate, or associate, with any of the current targets. Firstly, a product is taken across the columns to create a single column vector. A summation is then performed across the elements in this column vector to provide a single output value representing the MOL.

The MOL that is derived in (5.24) is of a similar form to that of the matrix permanent [145,146]. There are a range of techniques available for finding approximations of the permanent; approximations are required when the matrix is high-dimensional due to an exponential scaling in the number of operations necessary to compute it. This would be appropriate if we were to deal with very congested scenarios containing many targets and many sensor measurements. However, in the smaller surveillance cases presented in this thesis which have a relatively small number of measurements and targets, this likelihood or permanent can be computed directly.

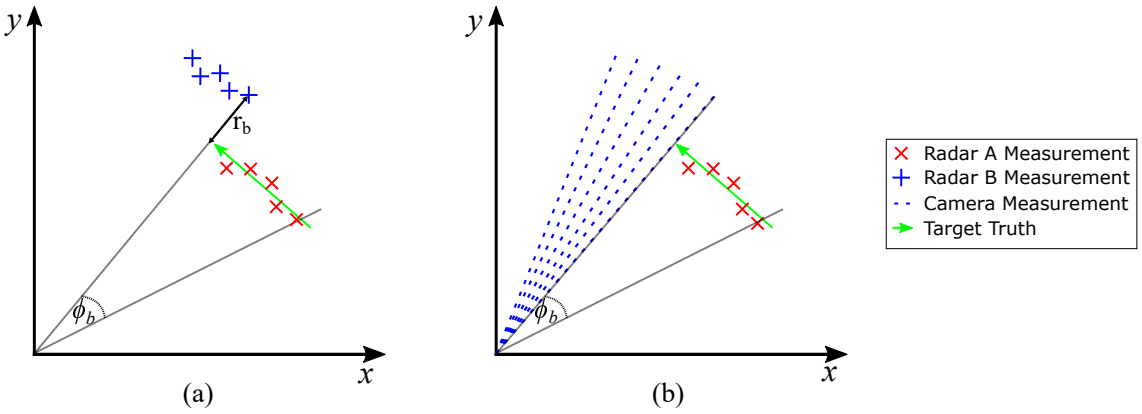


Figure 5.1: A reminder of the registration problems being resolved, as described initially in Chapter 1. (a) A homogeneous network containing two radars, with range bias r_b and azimuth bias ϕ_b. (b) A heterogeneous network where Radar B has been replaced with a camera that has azimuth registration bias ϕ_b.

5.3 Implementation

The two scenarios that the proposed method is aiming to solve are highlighted in Figure 5.1, using the implementation details presented next.

5.3.1 Sensor Registration

As before, the high-level process in the implementation will estimate the potentially time-varying sensor registration configuration $\mathbf{q}$ at each iteration of the algorithm. This will follow the parent process recursion given in Chapter 3.

Similar to the implementation in Chapter 4, the registration configuration $\mathbf{q}_k$ will be represented using a particle distribution such that $\mathbf{q}_k^i, 1 \leq i \leq N \approx \mathbf{q}_k$, where each i represents a different configuration. Each of the particles has its own weight w_k^i, and a corresponding set of underlying MTT statistics θ_k^i which will be dependent on the type of algorithm used in the low-level offspring process. Each particle $\mathbf{q}_k^i$ is an independent test of the registration parameters; the uncertainty in each of these tests is not shared or reflected in any other test. Following from the equations presented previously in Section 2.2.2 on page 20, the equations for the

parent process recursion are:

$$\hat{P}_{k|k-1}(\mathbf{q}_k|\mathbf{Z}_{1:k-1}) = \sum_{i=1}^{N} w_{k|k-1}^{i}\delta(\mathbf{q}_k - \mathbf{q}_k^i), \tag{5.25}$$

$$\hat{P}_{k}(\mathbf{q}_k|\mathbf{Z}_{1:k}) = \sum_{i=1}^{N} w_{k}^{i}\delta(\mathbf{q}_k - \mathbf{q}_k^i), \tag{5.26}$$

$$w_{k|k-1}^{i} = \sum_{j=1}^{N} w_{k-1}^{j}\hat{P}(\mathbf{q}_k^i|\mathbf{q}_{k-1}^j), \tag{5.27}$$

$$w_{k}^{i} = \frac{w_{k|k-1}^{i}\hat{\ell}_k^i}{\sum_{j=1}^{N} w_{k|k-1}^{j}\hat{\ell}_k^j} \tag{5.28}$$

where $\hat{\ell}_k^i = \hat{\ell}_k(\mathbf{q}_k^i|\mathbf{Z}_k)$ is the MOL function. After updating the high-level weights $w_k^i, i = 1, \ldots, N$, the *effective sample size* (Equation (2.28) on page 23) is computed, in order to test for *particle degeneracy*. If the effective sample size is below a given threshold $\tau_{resample}$, a *systematic resampling* [65] strategy is performed on the particle distribution to increase their spread in the high-level state space. The decision to use systematic resampling over other more traditional resampling methods in this application is justified due to the shape of the posterior distribution in the high-level state space. This distribution tends to be very sharp; with a very localised peak at the true value; small errors in azimuth can lead to track inaccuracies in the order of hundreds of metres in some cases. These inaccuracies lead to large reductions in the MOL function, and therefore the particle weights of the outliers. When these particle weights were resampled using the simple random resampling method, it was noticed that the particle(s) with very high weight were being chosen many times, reducing the diversity of the particles.

5.3.2 Multi-Target Tracking

The low-level offspring process in the hierarchy continues to estimate the potentially time-varying multi-target state $\mathbf{X}_k \in \mathcal{X}$, which is dependent on the sensor registration configuration $\mathbf{q}_k$. This multi-target state will continue to be updated through the offspring process recursion in Chapter 3. The MTT process will evolve with a first-order Markov model $f_{k|k-1}(\mathbf{X}_k|\mathbf{X}_{k-1})$, which will continue to be a near-constant velocity (NCV) model [52] as in Chapter 4.

A similar particle BP algorithm to the one that is implemented here can be found in [22]; however the factor graphs are different, due to the asynchronous nature of the sensors considered here. The implementation in [22] assumes a synchronous network of sensors, meaning that it is possible to perform the correlation, association and update steps in parallel for all sensors, and these messages can all be passed back to the variable node $\mathbf{y}$. For the asynchronous case presented in this thesis, this

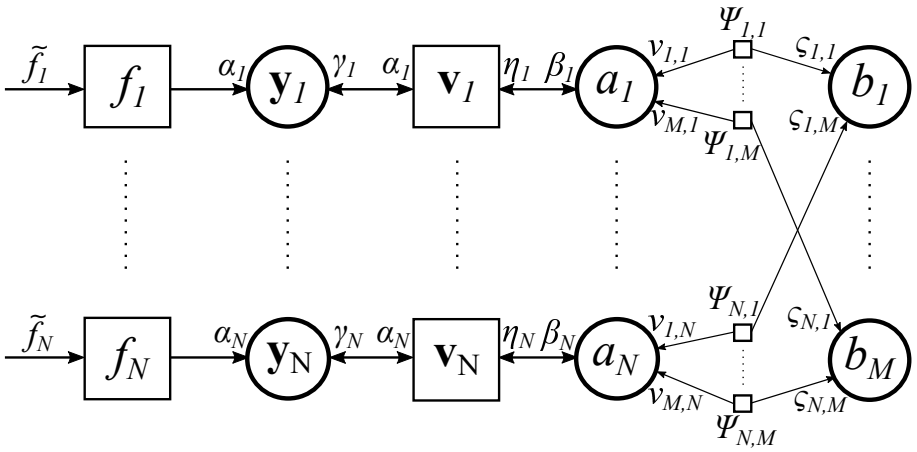


Figure 5.2: Factor Graph for MP for MTT implementation.

parallel, or batch processing method cannot be performed as only one sensor will provide measurements at a given iteration. The exact factor graph (assuming the beliefs in the prior factorise) used here is given in Figure 5.2, and contains the following notation:

$\tilde{f}$ **:** target beliefs from time $k-1$;

f **:** represents the prediction step $f(\mathbf{x}_{k,n}, r_{k,n}|\mathbf{x}_{k-1,n}, r_{k-1,n})$;

α **:** the marginal predictions;

v **:** likelihood function computed through (5.3) and (5.4);

β **:** contains correlation information to initialise the association process;

η **:** result of the association process;

γ **:** contains measurement update information.

All of the messages being passed around the graph are only *approximations* of the true messages because of the use of particle BP [22, 23].

5.4 Simulations

The results shown in the previous chapter were generated using the single-cluster framework with a grid-based method [49] to represent the parent process. The particles were spread evenly across a grid with rigid boundaries, giving a consistent parameter test, and removing the need for performing any sort of particle resampling [65]. This was overly restrictive as time was used to update particles that were far away from the true value, and not contributing to the overall result. In order to resolve this, the systematic resampling strategy is now introduced into the implementation in this chapter. Both of the presented simulations use $18^2 = 324$

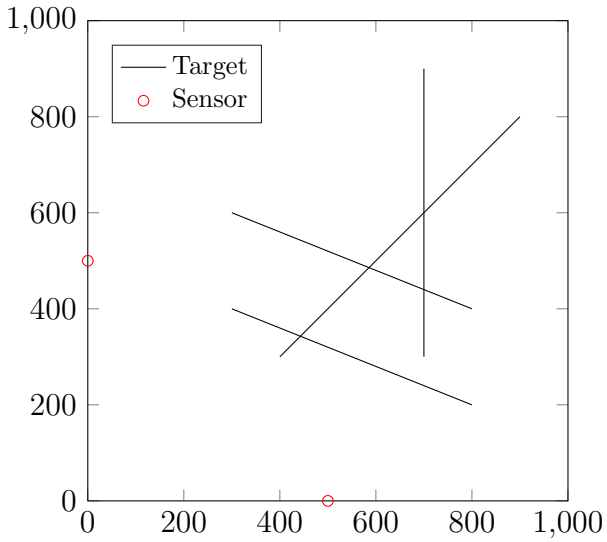


Figure 5.3: Simulated target trajectories for all scenarios. Sensor locations shown for homogeneous case; sensors are colocated at $(x, y) = (0, 0)$ in heterogeneous case.

particles in the parent distribution, and are initialised uniformly across an appropriate parameter space, as shown in Figure 5.4. The reasoning behind the choice of 324 particles, a square number, in this case is to have a uniform distribution across the two-dimensional space created for the multi-radar scenario. While this choice of uniform distribution follows the one used in the previous chapter, there is a lot of design flexibility available in the initial prior. The simulated parameters used in the offspring process are given in Table 5.2, and are kept consistent across all of the scenarios, and the 100 Monte Carlo (MC) runs. As shown in Figure 5.5, the results from a stability experiment show that the Generalised Optimal Subpattern Assignment (GOSPA) results should be consistent at 100 MC runs. An overview of the simulated target trajectories is shown in Figure 5.3. The dynamical model for the targets, and the sensor measurement models are the same as those presented in Section 3.4. To make a suitable comparison between two particle-based methods, the offspring process for the PHD method presented in this chapter is formed of the SMC-PHD filter [99], rather than the Extended Kalman Filter (EKF)-PHD or Panjer filters from Chapter 4.

The algorithms will also be compared in terms of the computation time per iteration. All results have been generated on a desktop PC containing an Intel Core i7-6700K CPU with a clock speed of 4 GHz and 16GB of RAM. Because all of the particles in the parent process are independent of one another, it is possible to further improve the performance as the estimation process can be parallelised. The filtering recursions for each of the particles are performed in parallel, and once completed, the normalisation of the particle weights in the parent process, and the resampling strategy can be performed sequentially.

The plots given in Figure 5.5 indicate that both the PHD approach and the MP

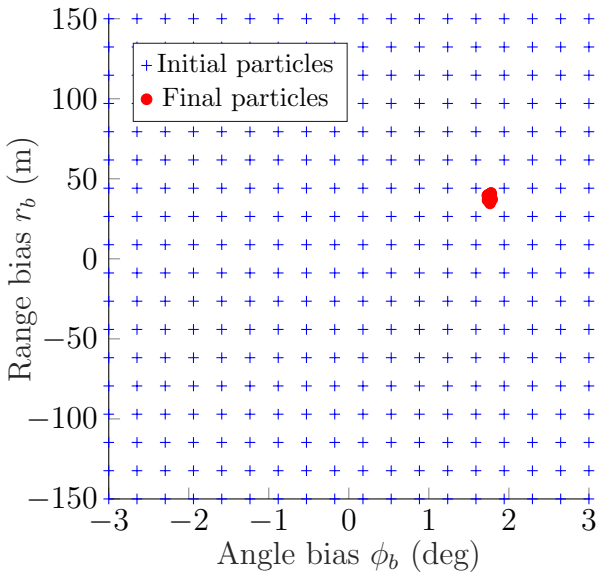


Figure 5.4: An example of the sensor configurations represented by a particle distribution in the parent process for the homogeneous scenario.

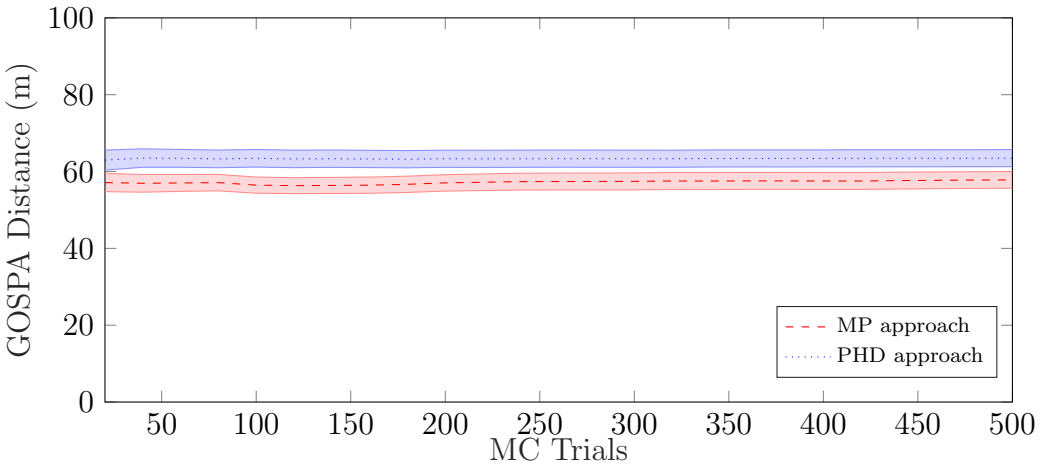


Figure 5.5: Indication of stability for the algorithms being tested.

Table 5.1: Measurement Noise Levels

Noise Level	σ_{r_r}	σ_{ϕ_r}	σ_{ϕ_c}
Low Noise	5 m	0.05°	0.01°
Medium Noise	10 m	0.1°	0.03°
High Noise	20 m	0.2°	0.05°

Table 5.2: Tracking Parameters

Quantity	Symbol	Value
Survival Probability	p_{s}	0.95
Gating Threshold	τ_{gate}	99%
Pruning Threshold	τ_{prune}	0.001
Extraction Threshold	$\tau_{extract}$	0.5
Resampling Threshold	$\tau_{resample}$	162
False Alarm Rate	λ_r, λ_c	2, 5
Birth Intensity	μ_b	1
Acceleration Noise	u	1 $m^2 s^{-3}$
Particles per Target	N	1000

approach are stable, even for low numbers of MC trials. The average target tracking accuracy given by GOSPA is consistent, and there is a small reduction in the spread of the final result as the number of trials increases. The spread reduces by 2.22 m for the PHD approach, and by 2.18 m for the MP approach.

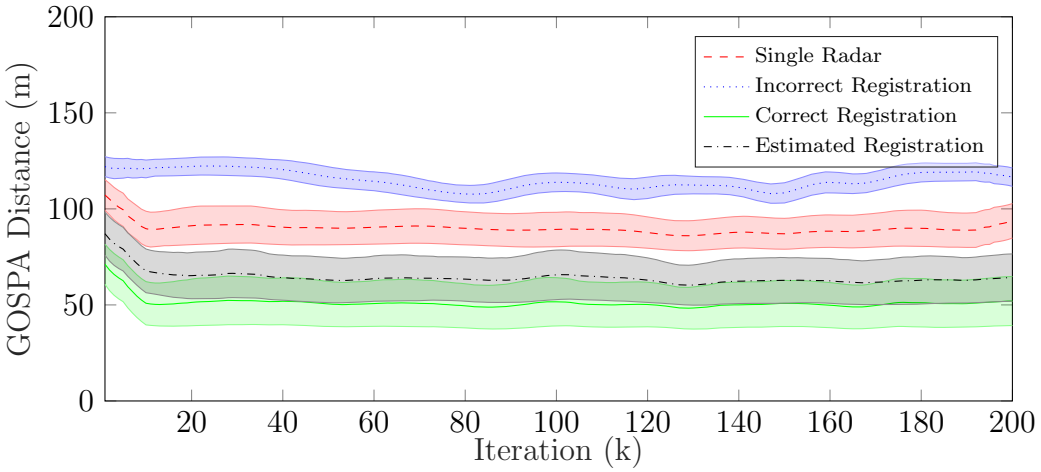
Results have been generated for three differing levels of measurement noise. In practical fusion systems, there are likely to be sensors present in the network that have different levels of measurement accuracy, which may also be time-varying. For example, the radar measurement uncertainty is often dependent on whether advanced signal processing techniques such as monopulse are available, and the angle off the radar boresight. Angles further away from the boresight, closer to the edge of the radar beam typically have a higher uncertainty than those closer to the centre [36]. The range uncertainty is dependent on factors such as the size of each range bin, and the type of radar waveform being exploited.

In all of the plots from Figure 5.6 to Figure 5.12, the results shown are from the low measurement noise case. A full summary of results for the low, medium and high noise cases (measurement noise values for these cases are shown in Table 5.1) are shown in Table 5.3 and Table 5.4.

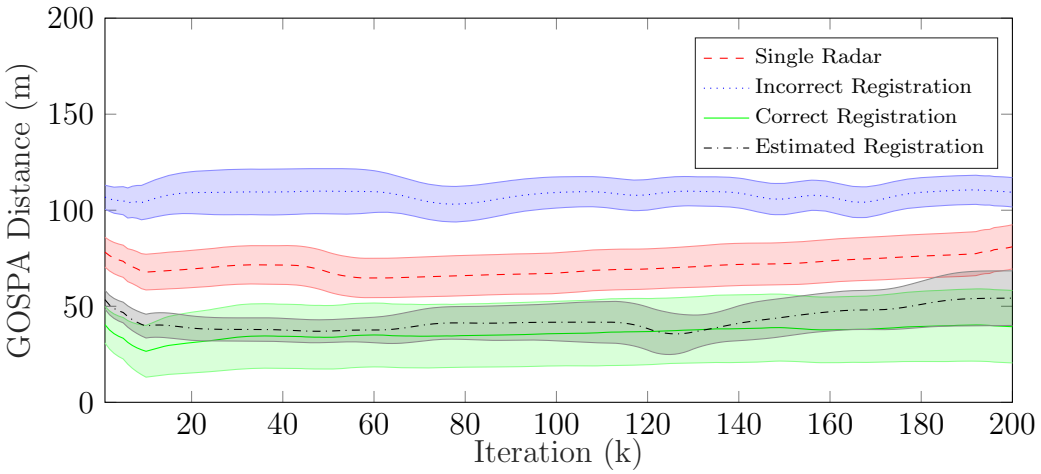
5.4.1 Multi-Radar

For this first scenario, a multi-radar sensor setup is considered, where two radars survey a common region of interest. The two radars themselves are physically separated by a number of kilometres in fixed and known locations; however there is some uncertainty in their relative azimuth orientation, and a range bias between the two sets of measurements. Both of these parameters will be estimated alongside the multiple target states. For this simulated scenario, the true biases simulated are $\mathbf{q} = [30\,\mathrm{m},\ 2^{\circ}]$.

From looking at Figure 5.6a and Figure 5.6b, it can be seen that the proposed

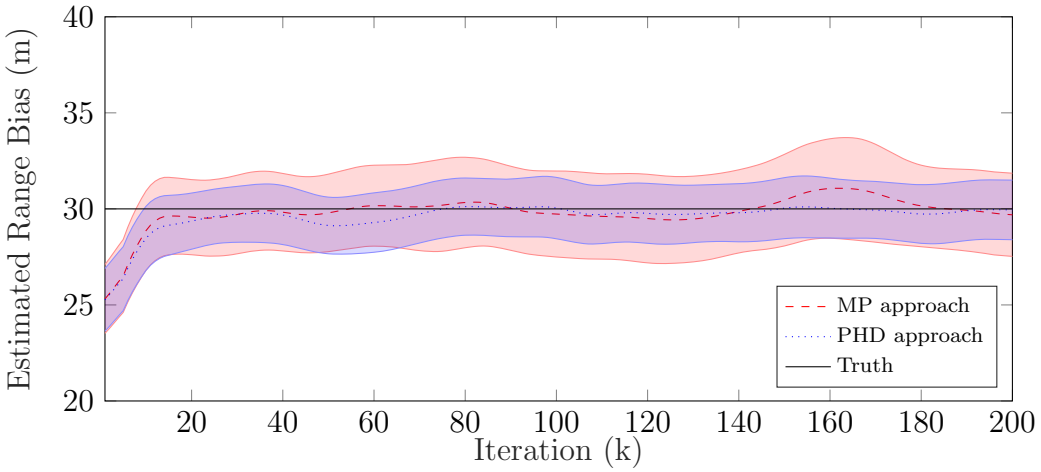


(a) PHD approach

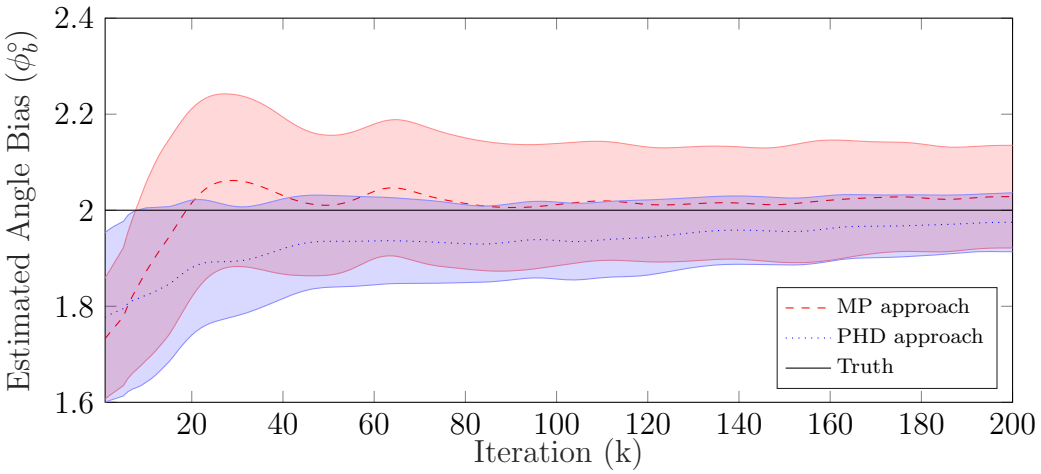


(b) MP approach

Figure 5.6: Multi-radar, $p_d = 0.99$

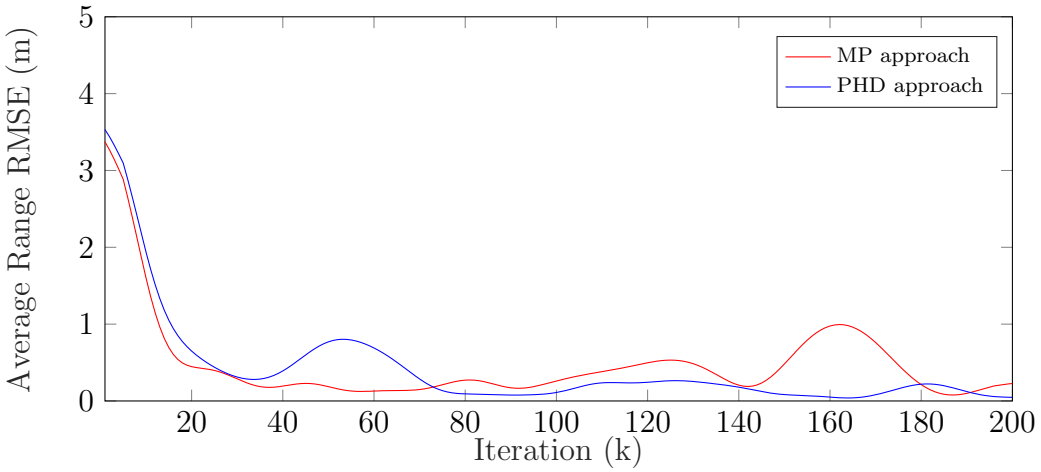


(a) Range estimation

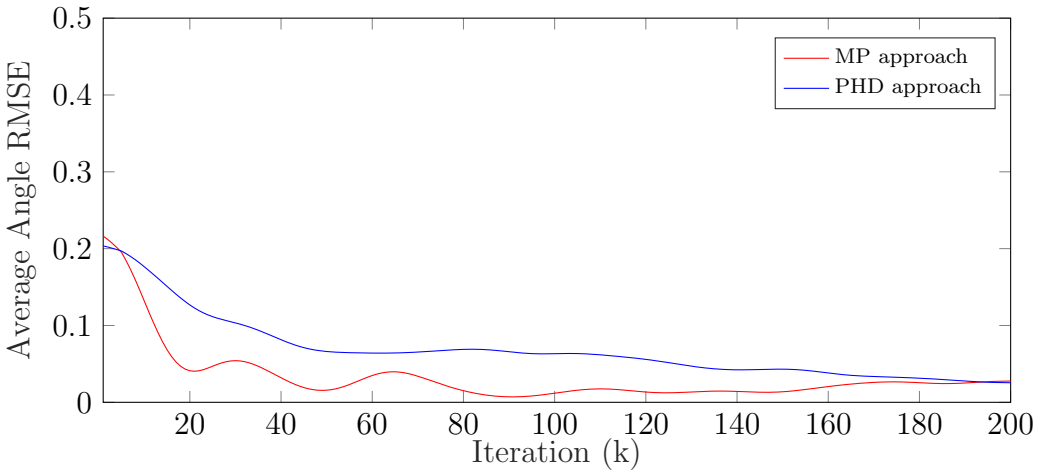


(b) Azimuth angle estimation

Figure 5.7: Parameter estimation, multi-radar, $p_d = 0.99$

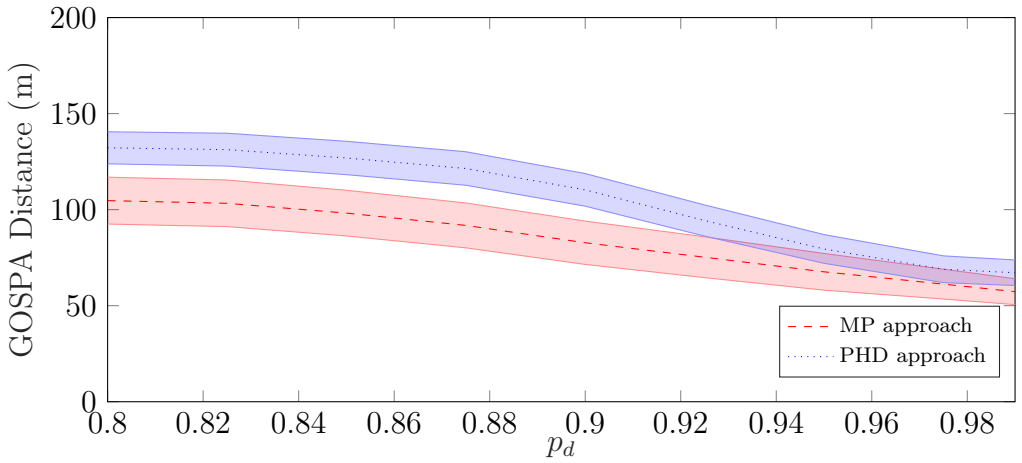


(a) Range Root Mean Square Error (RMSE)

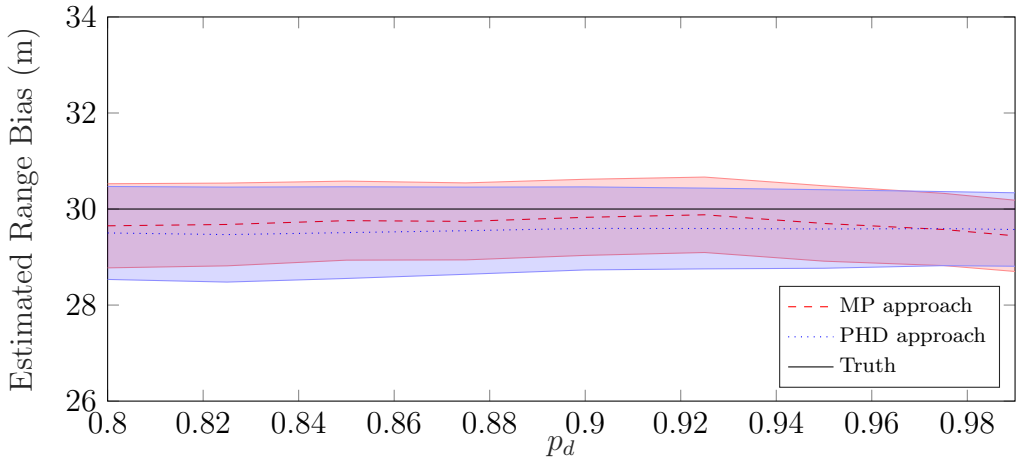


(b) Azimuth RMSE

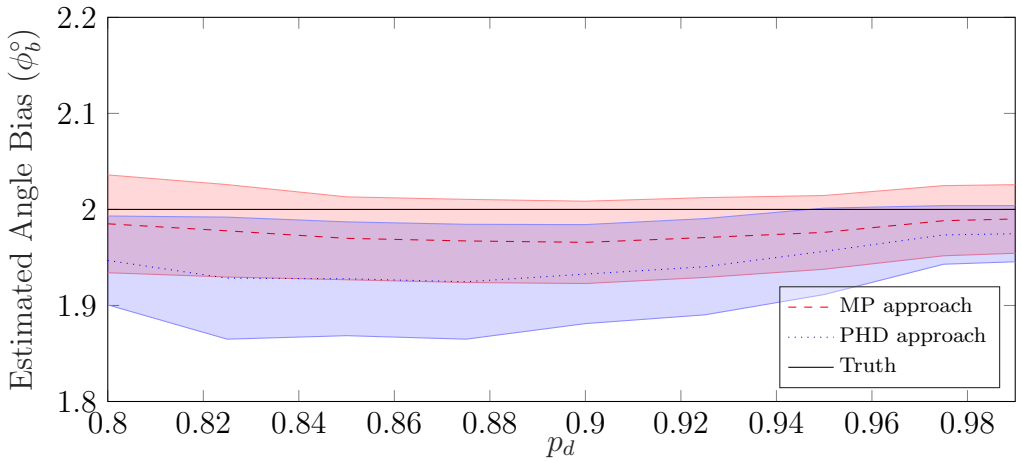
Figure 5.8: RMSE, multi-radar, $p_d = 0.99$



(a) GOSPA Distance



(b) Range estimation



(c) Azimuth angle estimation

Figure 5.9: Varying p_d results, homogeneous sensors, average value at $k = 200$.

method of simultaneous registration and tracking (shown as the black dot-dashed plot) outperforms the use of a single radar (shown as the red dashed plot) in terms of the GOSPA distance by 29.54 m and 26.73 m respectively, but does not reach the "optimal" performance of the perfectly registered set of sensors (shown in green). The PHD approach is 12.22 m away from the accuracy with correct registration, and 14.87 m away from the correct registration in the MP approach. As with the results shown and conclusions taken from Chapter 4, the case containing the unregistered set of sensors (shown as the blue dotted plot) performs much worse than the other cases, further emphasising the importance of having accurate registration to perform sensor fusion. There appears to be a larger gap between the single radar plot and unregistered plot in Figure 5.6b than in Figure 5.6a, which may be due to the sensor measurements appearing too far away from each other and therefore not correlating and associating with one another. For the PHD approach in Figure 5.6a, the reduction in accuracy between the single radar case and the unregistered case is 22.74 m, and for the MP approach the reduction is 28.37 m. Because the PHD approach from Chapter 4 does not attempt to resolve the DA problem explicitly, it may be more forgiving of outliers and therefore make loose measurement-to-target associations, hence reducing the GOSPA distance.

At the end of the scenario, the GOSPA distance for the PHD approach is 64.2 m and 54.2 m for the MP approach, giving a performance gain of approximately 10%. In terms of the registration estimation results shown in Figure 5.7, we see that the PHD approach has less variation than the MP approach in estimating the range bias (Figure 5.7b), however both methods are within ± 2 m of the true value for the majority of the scenario. The angle estimates in Figure 5.7b are consistently accurate throughout the scenario for the MP approach with very little variation. The average angle estimate from the MP approach is within 0.07° of the true value, and 0.13° from the true value for the PHD approach.

The plots in Figure 5.9 show results involving a range of probabilities of detection p_d. As the probability of detection increases from 0.80 to 0.99, the tracking accuracy continually increases for both methods, and the GOSPA distance decreases as expected; with the MP approach always outperforming the PHD approach. From the results involving the parameter estimation in Figure 5.9b and Figure 5.9c, there is only a small change in the average estimates for the parameters over p_d; however the spread of the result slowly decreases as p_d increases.

From the tabulated results in Table 5.3, which contain GOSPA values on a range of measurement noise levels, and a range of probabilities of detection, we can see that the MP method is more accurate at the low measurement noise level in almost all cases. In the medium noise cases, we start to see a decrease in accuracy for both methods, with the MP approach deteriorating at a faster rate than that of the PHD approach. This may be a result of the MP approach attempting to resolve the

data association problem explicitly; measurements may appear much further away from the target states, and therefore lie outside the correlation gates or thresholds. Because a measurement is outwith the threshold, it will not be eligible for association and it would be likely that a track is not updated and not declared as a target state for extraction. In the high noise scenario, both methods perform very poorly, as would be expected.

In terms of execution time, the PHD approach took 0.781 s per iteration on average, with the MP approach marginally slower at 1.086 s per iteration.

5.4.2 Heterogeneous Sensors

We now consider the heterogeneous sensing case, where a radar and an Infrared Search and Track (IRST) system are co-located on the same static platform, similar to the simulation presented in Chapter 4. The high update rate of the IRST system is exploited to update the track estimates more often. In this case, it is desired to estimate the relative angular bias $q = 2°$ between the sensors, alongside the multiple target states.

From Figure 5.10a, it can be seen that both the PHD approach and the MP approach give better accuracy than their respective single radar cases in terms of the GOSPA metric, but not as accurately as their correct registration cases as expected. The approach that is proposed in this Chapter gives a higher accuracy than the PHD approach from Chapter 4 in almost all of the low measurement noise cases. At the end of the scenario ($k = 200$), the GOSPA distance for the MP approach is 47.8 m, and 72.6 m for the PHD approach, giving a more substantial performance boost than that shown in the multi-radar scenario (Section 5.4.1). The parameter estimation result given in Figure 5.11a shows that the two methods are closely comparable in terms of accuracy, with the MP approach 0.05° away from the true value on average, and the PHD approach 0.06° away on average.

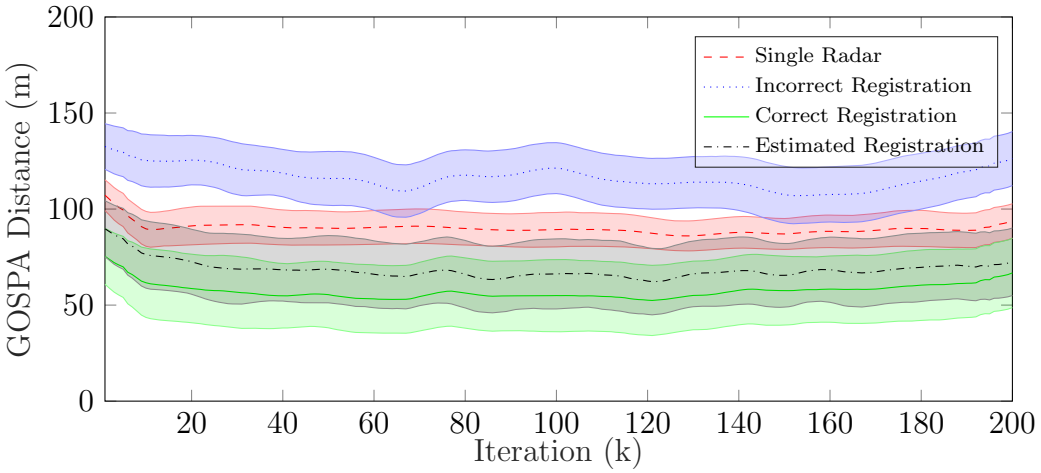
When considering the results over a range of probabilities of detection in Figure 5.12, a similar behaviour to that seen in Figure 5.9 is apparent; the GOSPA distance for both methods decreases while p_{d} increases, with the MP approach more accurate than the PHD approach; and the MP approach provides a more accurate average estimate of the bias angle between the sensors. The full breakdown of the results on the range of probabilities of detection, and the different measurement noise levels is given in Table 5.4. Again, we see very similar behaviour to that of the multi-radar results; the proposed MP method performs most accurately in the low noise cases, the MP approach deteriorates more rapidly than the PHD approach when considering the medium noise level, and both methods perform poorly at the high noise level as expected. The PHD approach took on average 1.203 s per iteration to execute, with the proposed MP approach taking 1.530 s per iteration to execute.

Table 5.3: Homogeneous Network, Average GOSPA Distances at $k = 200$

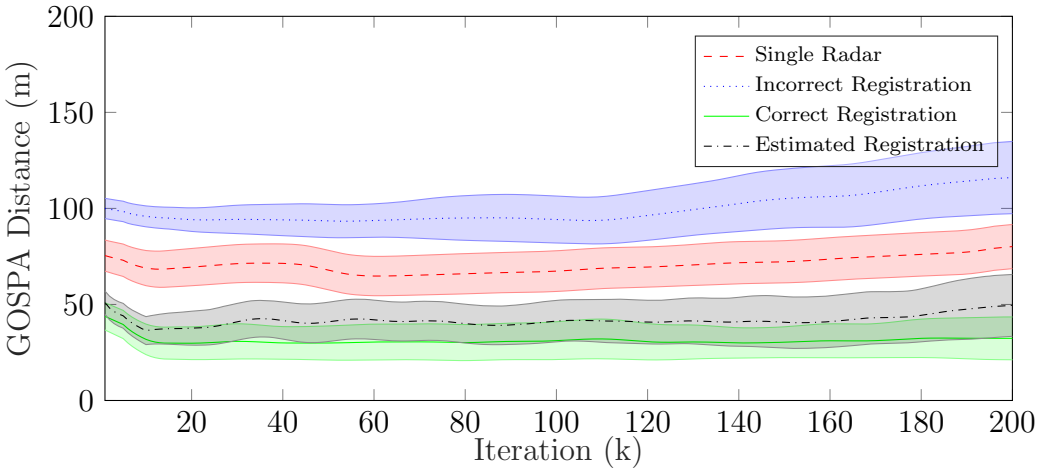
	p_d	0.80			0.85			0.90			0.95			0.99		
	Noise	Low	Med	High	Low	Med	High	Low	Med	High	Low	Med	High	Low	Med	High
Single Radar	PHD	**144**	164	202	**135**	155	197	**122**	144	193	**101**	132	189	**94**	126	187
	MP	**120**	192	200	**117**	189	199	**106**	186	197	**102**	178	195	**80**	167	186
Incorrect	PHD	**152**	191	209	**145**	166	207	**137**	150	203	**117**	136	197	**117**	134	197
	MP	**155**	199	205	**144**	195	203	**139**	191	203	**128**	188	208	**109**	182	203
Proposed	PHD	**134**	164	183	**129**	145	195	**115**	133	191	**73**	98	170	**64**	95	151
	MP	**107**	189	199	**101**	182	195	**83**	170	179	**69**	149	158	**54**	136	154
Correct	PHD	**130**	150	164	**121**	140	181	**103**	130	178	**62**	82	158	**52**	78	139
	MP	**97**	183	194	**87**	180	190	**62**	156	176	**58**	140	165	**39**	134	150

Table 5.4: Heterogeneous Network, Average GOSPA Distances at $k = 200$

	p_d	0.80			0.85			0.90			0.95			0.99		
	Noise	Low	Med	High	Low	Med	High	Low	Med	High	Low	Med	High	Low	Med	High
Single Radar	PHD	**144**	164	202	**135**	155	197	**122**	144	193	**101**	132	189	**94**	126	187
	MP	**120**	192	200	**117**	189	199	**106**	186	197	**102**	178	195	**80**	167	186
Incorrect	PHD	**170**	180	219	**164**	173	213	**155**	167	211	**135**	151	203	**126**	142	197
	MP	**189**	216	222	**175**	211	222	**160**	207	220	**140**	204	218	**116**	193	210
Proposed	PHD	**137**	152	192	**130**	145	186	**116**	136	182	**86**	116	175	**73**	103	168
	MP	**114**	189	197	**91**	187	194	**79**	182	196	**49**	174	191	**48**	161	182
Correct	PHD	**127**	141	181	**118**	132	177	**108**	124	174	**79**	105	165	**67**	93	157
	MP	**101**	179	193	**84**	177	192	**56**	180	193	**44**	171	187	**32**	159	181

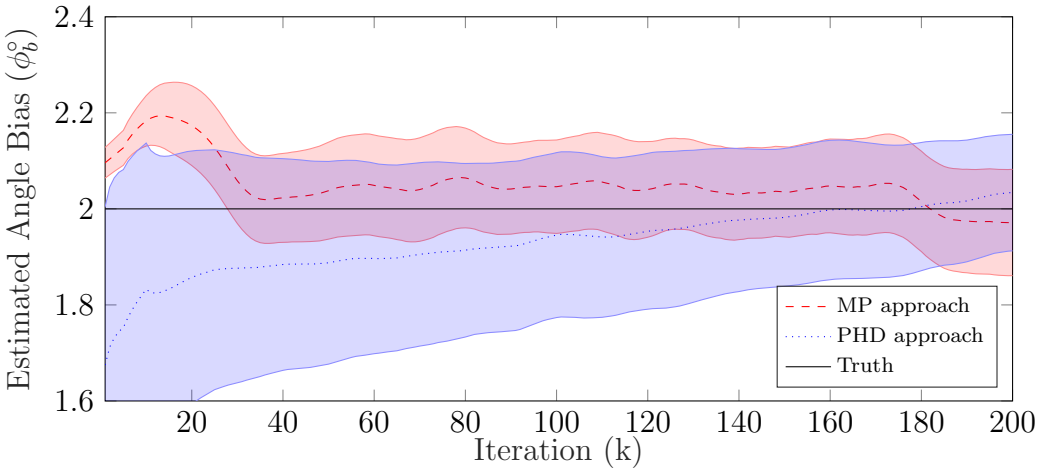


(a) PHD approach

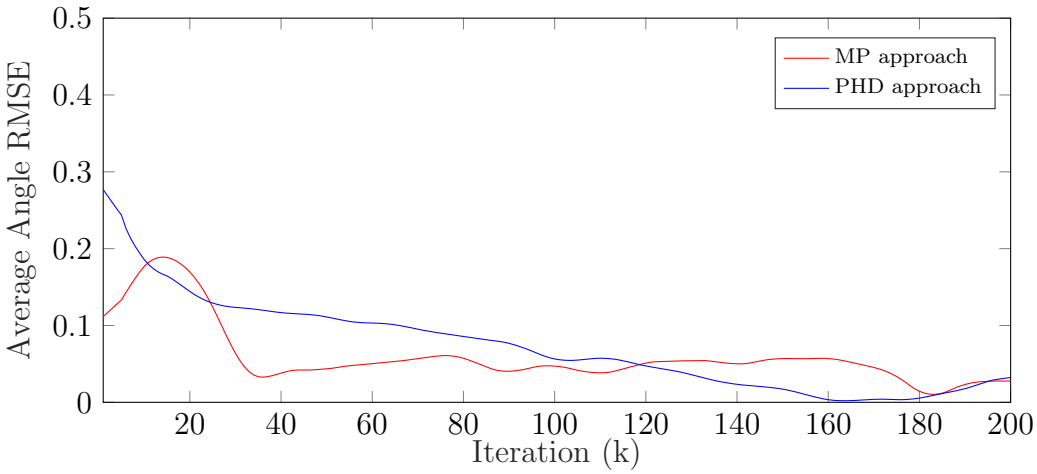


(b) MP approach

Figure 5.10: Heterogeneous sensors, $p_d = 0.99$

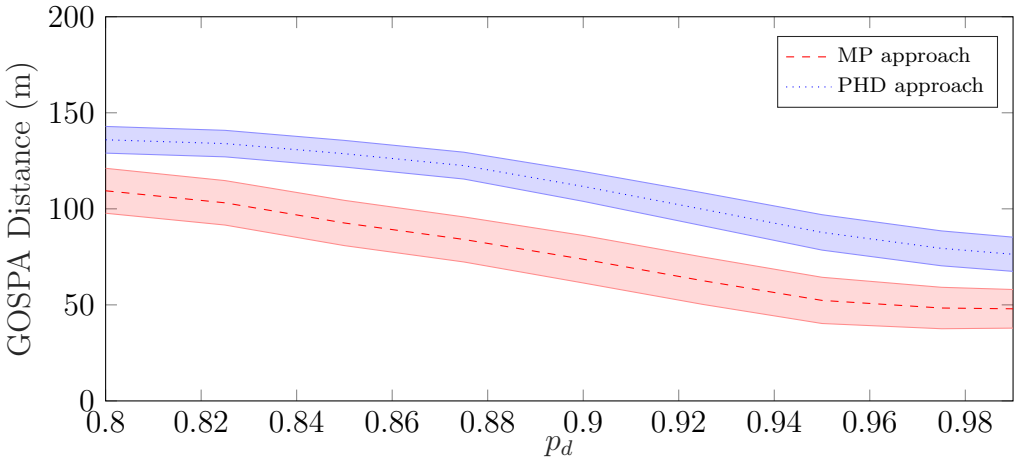


(a) Azimuth angle estimation

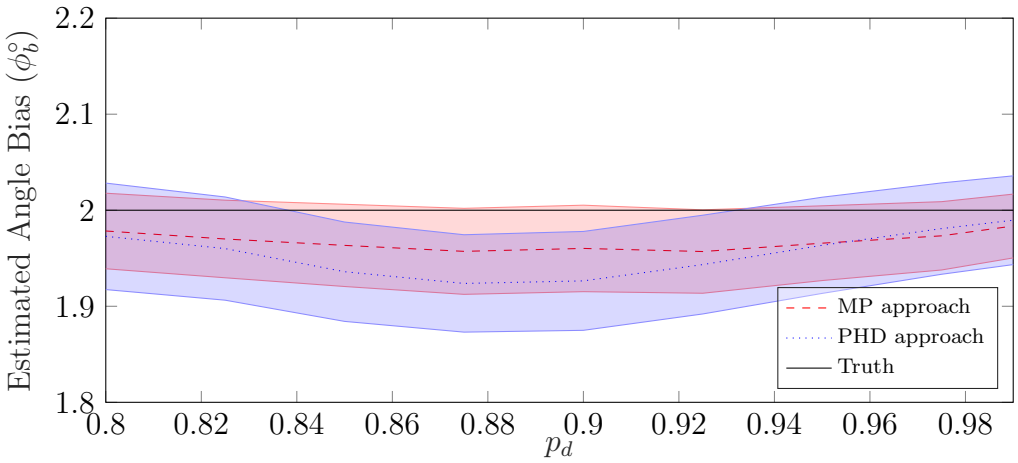


(b) Azimuth RMSE

Figure 5.11: Angular bias estimation, heterogeneous sensors, $p_d = 0.99$



(a) GOSPA Distance



(b) Azimuth angle estimation

Figure 5.12: Varying p_d results, heterogeneous sensors, average value at $k = 200$.

5.5 Conclusions

From the comprehensive simulations presented in the previous section, it can be seen that the proposed approach of using an MP algorithm inside the single-cluster method framework gives more accurate target tracking and sensor registration estimation than the point process method presented in Chapter 4. When considering the low noise cases for both sensor setups, the MP approach performs approximately 17% more accurately in terms of the GOSPA metric. Again the results also highlight the importance of maintaining an accurately registered set of sensors when attempting to perform data fusion, as tracking accuracy is greatly reduced when the correct registration is not taken in to account.

When considering the medium measurement noise and high measurement noise cases, the results show that the MP-based method proposed in this chapter deteriorates in performance at a faster rate than the PHD approach. By having these larger values of noise included in the sensor measurements, the measurements are more likely to be reported much further away from the true target location. When these are used in the correlation step of the MP method, it is quite likely that they will lie outside of the gating threshold, and therefore, will not be associated with existing targets. However, as the PHD approach does not attempt to resolve data association, it may be more forgiving to these "outlier" measurements and hence improve the tracking accuracy.

Previous works in the area of MP for MTT have shown that it should be less computationally expensive to use MP methods, rather than point process methods such as the PHD filter [22, 23]. However so far, this has not been seen in this work; the MP implementation is currently approximately 20% slower than that of the PHD approach. This may be due to coding inefficiencies, and problems relating to parallelisation of the algorithm such as reading/writing overheads.

Now that we have a more accurate solution to resolving the sensor registration and fusion problem for the two-sensor case, the next chapter of this thesis will attempt to scale the algorithm in to larger scenarios that contain many more sensors.

Chapter 6

Single-Cluster Methods vs. BP Frameworks in Larger Networks

So far, the use of *single-cluster methods* for performing joint registration and fusion has proven to be a suitable solution. It has been shown in both Chapter 4, and in Chapter 5 that the proposed methods have been able to accurately estimate the sensor registration parameters, and the states of the multiple targets. Chapter 5 proposed the idea of inserting Belief Propagation (BP) algorithms in to the single-cluster framework, and the results showed an increase in tracking and registration estimation accuracy for scenarios containing two sensors, over the implementation based on point process methods from Chapter 4. However, one aspect of performance that is important for practical deployment has not yet been explored; the *scalability* of these algorithms, how many sensors could realistically be registered, and their measurements fused, in a larger sensor network?

A very recent addition to the literature [50] has proposed a detailed factor graph and BP implementation that allows for other parameters to be estimated alongside the multiple target states. This method however, as with many other pieces of literature discussed in Chapter 2, is also based on assumptions such as synchronous sensors, and that the parameter estimates must be chosen from a pre-determined, discrete set of values. This chapter will explore the possibility of extending this algorithm to estimate appropriate sensor registration parameters alongside the target states. The proposed implementation will then be compared to the hierarchical model that has been developed previously in Chapter 4 and Chapter 5 of this thesis. The simulated scenarios will begin at two sensors, and be increased to a maximum of 16 sensors. The results presented should give a perspective of how well these algorithms could perform when deployed in a real system.

The main contributions of this chapter, and advancements from Chapter 4 and Chapter 5, include:

- the extension of the previously proposed methods, to address much larger sets of sensors that observe a common region, in order to test the scalability of the

methods presented;

- the extension of the implementation given in [50], by showing that it is possible to estimate from a continuous space containing variables such as distances and orientation angles, rather than being constrained to selecting from finite set of discrete values;

- a robust and well-rounded comparison between the *single-cluster, hierarchical approach* presented in Chapter 4 and Chapter 5, and the *full BP-based approach* developed during this Chapter.

6.1 Joint Estimation using Message Passing

In previous chapters, only the hierarchical estimation framework has been considered as a solution to the joint sensor registration and fusion problem. In order to implement the full-Message Passing (MP) approach for parameter estimation, the factor graph that was presented in Chapter 5 will need to be extended, so that the registration parameters can now be represented with messages and beliefs, rather than using the parent process. These new messages will still be represented using particles, but will be embedded into the graph structure, and not predicted and updated through the parent process recursion in equations (3.1) and (3.2) on page 58.

The formulation given in [50] will need to be adapted to suit this problem. The parameters should not be explicitly chosen from a set of pre-defined values; the particles should search the parameter space for the best set of parameters. Aspects of the single-cluster, or hierarchical, method from Chapter 5 will be implemented, such as the particle resampling strategy for the parameter estimation. The initial parameter hypotheses are distributed uniformly across an appropriately sized parameter space, and these will evolve independently through a first-order Markov model. The notion of having a reference sensor will still apply in this Chapter; all other sensors will be registered with respect to this single reference. There will be one appropriately sized parameter space for each of the unregistered sensors, which will be predicted and updated separately. This proposed method does not stack all of the registration parameters for all sensors into a common vector to try and resolve the *global registration* between all pairs of sensors.

With the parameter hypotheses represented with particles, and the registration parameters for a sensor s being independent of all other sensors, the prior Probability Mass Function (PMF) of the estimated parameters will be

$$p(\mathbf{q}) = \prod_{s=1}^{S-1} p\left(\mathbf{q}_0^{(s)}\right) \prod_{k'=1}^{k} p\left(\mathbf{q}_{k'}^{(s)} | \mathbf{q}_{k'-1}^{(s)}\right). \tag{6.1}$$

where $p\left(\mathbf{q}_{k'}^{(s)}|\mathbf{q}_{k'-1}^{(s)}\right)$ is an appropriate first-order Markov model to represent the change in the registration parameters over time, and $s \in \{1, \ldots, S-1\}$ represents the sensor number. We only require up to sensor $S-1$ as registration is not required on the reference sensor.

The posterior probability density function (pdf) of the target states $f(\mathbf{x}_{k,n}, r_{k,n}|\mathbf{Z})$ is a marginal of the joint posterior pdf $f(\mathbf{X}, \mathbf{r}, \mathbf{a}, \mathbf{b}, \mathbf{q}|\mathbf{Z})$ that includes all of the target states $\mathbf{X}$, the existence variables $\mathbf{r}$, the association variables $\mathbf{a}$ and $\mathbf{b}$, the registration parameters $\mathbf{q}$, and the measurements $\mathbf{Z}$. A detailed factor graph that represents the factorisation of the joint posterior pdf $f(\mathbf{X}, \mathbf{r}, \mathbf{a}, \mathbf{b}, \mathbf{q}|\mathbf{Z})$ is required so that an appropriate MP schedule and order can be defined. Because $\mathbf{Z}$ is observed by the sensors and is fixed, the number of measurements $\mathbf{m}$ will also be fixed. This then leads to the equations defined in [50, Eq's (16)-(18)]:

$$
\begin{aligned}
& f(\mathbf{X}, \mathbf{r}, \mathbf{a}, \mathbf{b}, \mathbf{q}|\mathbf{Z}) \\
& = f(\mathbf{X}, \mathbf{r}, \mathbf{a}, \mathbf{b}, \mathbf{q}, \mathbf{m}|\mathbf{Z}) \\
& \propto f(\mathbf{Z}|\mathbf{X}, \mathbf{r}, \mathbf{a}, \mathbf{b}, \mathbf{q}, \mathbf{m}) f(\mathbf{X}, \mathbf{r}, \mathbf{a}, \mathbf{b}, \mathbf{q}, \mathbf{m}) && (6.2) \\
& = f(\mathbf{Z}|\mathbf{X}, \mathbf{r}, \mathbf{a}, \mathbf{m}) f(\mathbf{X}, \mathbf{r}, \mathbf{a}, \mathbf{b}, \mathbf{q}, \mathbf{m}) && (6.3) \\
& = f(\mathbf{Z}|\mathbf{X}, \mathbf{r}, \mathbf{a}, \mathbf{m}) p(\mathbf{a}, \mathbf{b}, \mathbf{m}|\mathbf{X}, \mathbf{r}, \mathbf{q}) f(\mathbf{X}, \mathbf{r}) p(\mathbf{q}) && (6.4)
\end{aligned}
$$

6.2 Implementation

The overall algorithm follows many of the same assumptions and MP rules that were defined earlier in Chapter 5, such as messages always being sent forward in time, and iterative Message Passing is only performed for the data association function. The algorithm implemented here follows the same steps as the one presented in Section 5.1.1 starting on page 84, but now contains extra messages and computations to estimate the registration parameters on the factor graph. The steps carried out follow the same order, and start with the prediction step.

Each and every target still follows a near-constant velocity (NCV) motion model over time, whereas the registration parameters are expected to follow a random walk, or Brownian motion model. The parameters are assumed to be static over time; the inclusion of the random walk will allow for the variance of the particles to be increased a small amount, and help to avoid the problems of *particle degeneracy* and *particle impoverishment* from Section 2.2.2. We now have two separate messages that are the result of the prediction step, one representing the targets, and one to represent the parameters. The target message is given by

$$
\alpha(\mathbf{x}_{k,n}, r_{k,n}) = \sum_{r_{k-1,n} \in \{0,1\}} \int \tilde{f}(\mathbf{x}_{k-1,n}, r_{k-1,n}) \times f(\mathbf{x}_{k,n}, r_{k,n}|\mathbf{x}_{k-1,n}, r_{k-1,n}) \mathrm{d}\mathbf{x}_{k-1,n}, \tag{6.5}
$$

which is the same as equation (5.1). The new parameter message that represents the prediction of the registration parameters is given by

$$\chi\left(\mathbf{q}_k^{(s)}\right) = \tilde{p}\left(\mathbf{q}_{k-1}^{(s)}\right) p\left(\mathbf{q}_k^{(s)} | \mathbf{q}_{k-1}^{(s)}\right) \tag{6.6}$$

where $\tilde{p}(\mathbf{q}_{k-1}^{(s)})$ are the parameter estimates for sensor s at time $k-1$, and $p\left(\mathbf{q}_k^{(s)} | \mathbf{q}_{k-1}^{(s)}\right)$ applies the Brownian motion to the registration parameter states.

Moving to the correlation step, the calculation for the messages $\beta\left(a_{k,n}^{(s)}\right)$ has been extended from Chapter 5, such that it includes the messages representing the predicted parameter estimates from (6.6). These are now calculated as

$$\beta\left(a_{k,n}^{(s)}\right) = \sum_{r_{k,n}\in\{0,1\}} \int v\left(\mathbf{x}_{k,n}, r_{k,n}, a_{k,n}^{(s)}, \mathbf{q}_k^{(s)}; \mathbf{Z}_k^{(s)}\right) \times \alpha(\mathbf{x}_{k,n}, r_{k,n}) \chi\left(\mathbf{q}_k^{(s)}\right) \mathrm{d}\mathbf{x}_{k,n} \tag{6.7}$$

The data association step is identical to the previous implementation, and is based on Sum-Product Algorithm for Data Association (SPADA) [77], where iterated messages are passed until a stopping criteria is reached. This stopping criteria is still preset to a value of $P = 30$ iterations as in Chapter 5. The equations for the SPADA algorithm are given in equations (5.5) to (5.8).

Next, depending if a target continues to exist or not, the measurement update steps are given by

$$\gamma^{(s)}(\mathbf{x}_{k,n}, r_{k,n} = 1) = \sum_{\mathbf{q}_k^{(s)}} \sum_{a_{k,n}^{(s)}=0}^{M_k^{(s)}} v\left(\mathbf{x}_{k,n}, r_{k,n} = 1, a_{k,n}^{(s)}, \mathbf{q}_k^{(s)}; \mathbf{Z}_k^{(s)}\right) \times \chi\left(\mathbf{q}_k^{(s)}\right) \eta\left(a_{k,n}^{(s)}\right) \tag{6.8}$$

$$\gamma^{(s)}(\mathbf{x}_{k,n}, r_{k,n} = 0) \triangleq \eta\left(a_{k,n}^{(s)} = 0\right). \tag{6.9}$$

Equation (6.8) differs from equation (5.9), as it now also includes the messages relating to the predicted registration parameters; equation (6.9) is the same as equation (5.10).

Finally, updated beliefs need to be calculated for the multiple target states, and for the estimated parameters. The beliefs that approximate the posterior pdfs of the target states are calculated by

$$\tilde{f}(\mathbf{x}_{k,n}, r_{k,n}) = \frac{1}{C_{k,n}} \alpha(\mathbf{x}_{k,n}, r_{k,n}) \gamma^{(s)}\left(\mathbf{x}_{k,n}, r_{k,n}\right) \tag{6.10}$$

where the normalisation constants $C_{k,n}$ are equivalent to those given in equation (5.13). The final beliefs approximating the posterior pdfs of the registration parameters are given by

$$\tilde{p}\left(\mathbf{q}_k^{(s)}\right) = \chi\left(\mathbf{q}_k^{(s)}\right) \epsilon\left(\mathbf{q}_k^{(s)}\right) \tag{6.11}$$

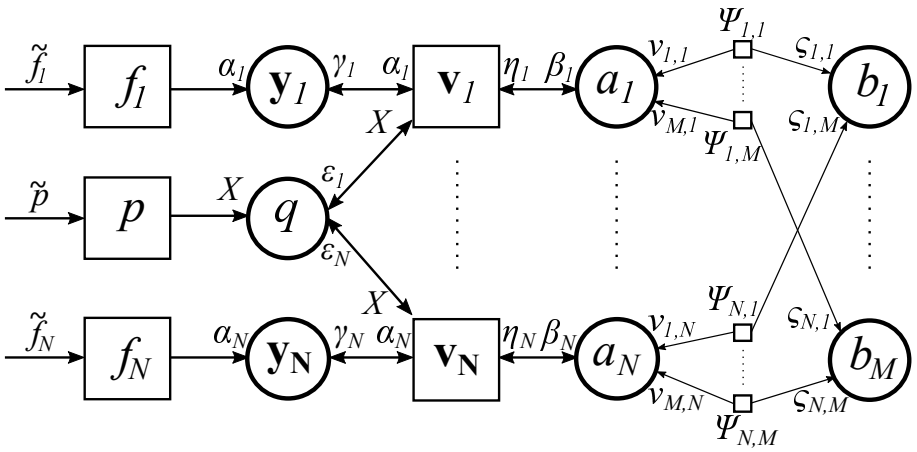


Figure 6.1: Factor Graph for fully-MP implementation.

with

$$\epsilon\left(\mathbf{q}_k^{(s)}\right) = \sum_{a_{k,n}^{(s)}=0}^{M_k^{(s)}} \int v\left(\mathbf{x}_{k,n}, r_{k,n}=1, a_{k,n}^{(s)}, \mathbf{q}_k^{(s)}; \mathbf{Z}_k^{(s)}\right) \eta\left(a_{k,n}^{(s)}\right) \times \alpha(\mathbf{x}_{k,n}, r_{k,n}=1)\mathrm{d}\mathbf{x}_{k,n} + \eta\left(a_{k,n}^{(s)}=0\right)\alpha(\mathbf{x}_{k,n}, r_{k,n}=0) \quad (6.12)$$

The exact factor graph used here is given in Figure 6.1. This factor graph does differ slightly from the graph proposed in [50]; as again one of the underlying assumptions made in the article is that the sensors all operate synchronously, and a batch update of all of the target states, and all parameters, can be undertaken.

6.3 Simulations

The simulated scenario used in this chapter is inspired in part by the scenarios presented by Uney et al. in [122, 123], where a number of sensors are placed in a grid formation and observe a number of targets moving around inside a small surveillance region. This scenario could potentially occur around a small site such as an airport for example. Although these articles consider a distributed fusion scenario, where track information is traded between neighbouring sensors, this simulation and subsequent results will assume a centralised fusion architecture as with previous chapters. This is also loosely based on the Sensing for Asset Protection with Integrated Electronic Networked Technology (SAPIENT) interface [124], where a number of plug-and-play sensors can be connected to a Fusion Centre (FC), also referred to as a High-Level Decision Making Module (HLDMM) in [124], and sensor registration and/or management performed. All of the sensors will produce range-bearing measurements following the radar measurement model in Chapter 3. The target trajectories are identical to those used in the simulations in Chapter 5, and an overview of this scenario is given in Figure 6.2.

Table 6.1: Tracking Parameters - Simulations

Quantity	Symbol	Sim Value
Detection Probability	p_{d}	0.99
Survival Probability	p_{s}	0.95
Gating Threshold	τ_{gate}	99%
Pruning Threshold	τ_{prune}	0.001
Merging Threshold	τ_{merge}	0.8
Extraction Threshold	τ_{extract}	0.5
False Alarm Rates	λ_r, λ_c	2, 5
Birth Intensity	μ_b	1
Acceleration Noise (ms^{-2})	u	1
Radar Measurement Noise (m,°)	σ_{r_r}, σ_{ϕ_r}	2, 0.01

The true range and azimuth bias in each of the sensors will be initialised to an integer value between $[-150\,\mathrm{m} \rightarrow 150\,\mathrm{m}]$ and $[-3^\circ \rightarrow 3^\circ]$ respectively. The particles used to estimate these biases will be initialised uniformly across this space, with all of these hypotheses being equally weighted. All of the results presented here are averaged over 100 Monte Carlo (MC) runs, where in each run, the sensor measurements generated are randomised, however the range and azimuth biases will be kept consistent. The registration parameter estimation will be compared using an average Root Mean Square Error (RMSE), with the target tracking accuracy again compared with the Generalised Optimal Subpattern Assignment (GOSPA) metric [128]. The parameters used in the GOSPA metric are $p = 2, c = 100\,\mathrm{m}$ and $\alpha = 2$.

In the following results, the proposed method from Chapter 5 is referred to as *Hierarchical Models*, and the method proposed earlier in this chapter is referred to as *Fully MP*.

6.3.1 Experiment 1: Two Sensor, Varying Hypotheses

The first experiment again considers the two-sensor case, with two radars detecting targets. This will test the scalability of the algorithms in terms of the number of parameter hypotheses required to accurately register the sensors and mitigate biases. The number of particles used to estimate the parameters will be increased through the square numbers from $2^2 \rightarrow 20^2$.

From Figure 6.3, it can be seen that there is a little difference between the two methods over time; the hierarchical approach appears to reach the true azimuth bias faster than the fully MP approach, and therefore reaches a lower GOSPA distance at an earlier iteration. The additive range bias estimates shown in Figure 6.3a are comparable across the whole scenario. After approximately iteration $k = 60$, the

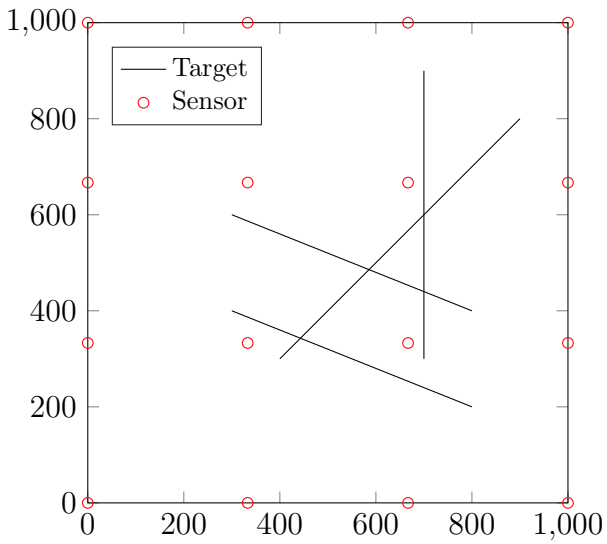


Figure 6.2: Simulated scenario containing the maximum 16 sensors.

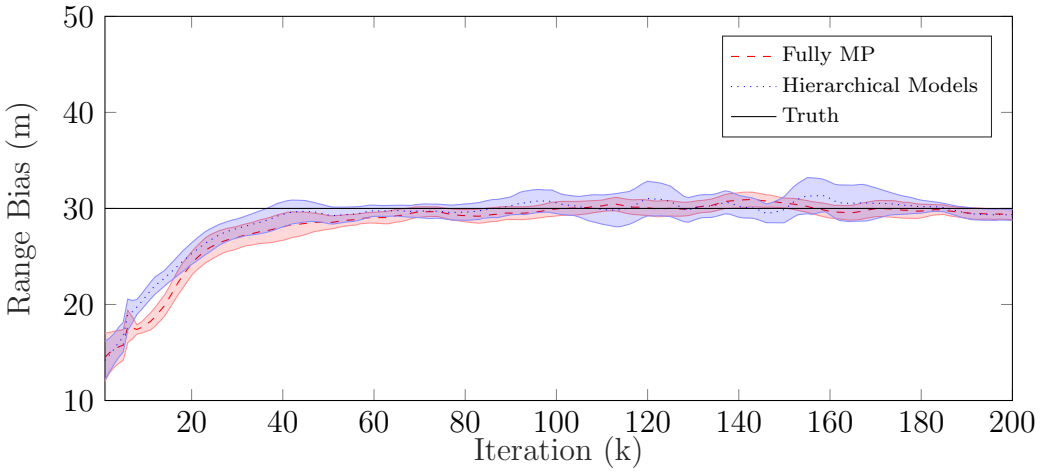
GOSPA distances in Figure 6.3c are almost identical. Looking at the RMSE results in Figure 6.4, there appears to be a curve on the performance. Using 100 hypotheses to represent the registration parameters appears to give an "optimal" trade-off between accuracy and efficiency; using any more hypotheses than this would expend more computational resources with no further gains in tracking accuracy, or registration estimation accuracy.

An important result is shown in Figure 6.5, where the average iteration execution times are compared. Both techniques scale linearly in the number of hypotheses as expected; however the hierarchical model follows a much steeper gradient than that of the fully-MP approach. The fully-MP approach is approximately 2.25 times faster on average across all of the runs.

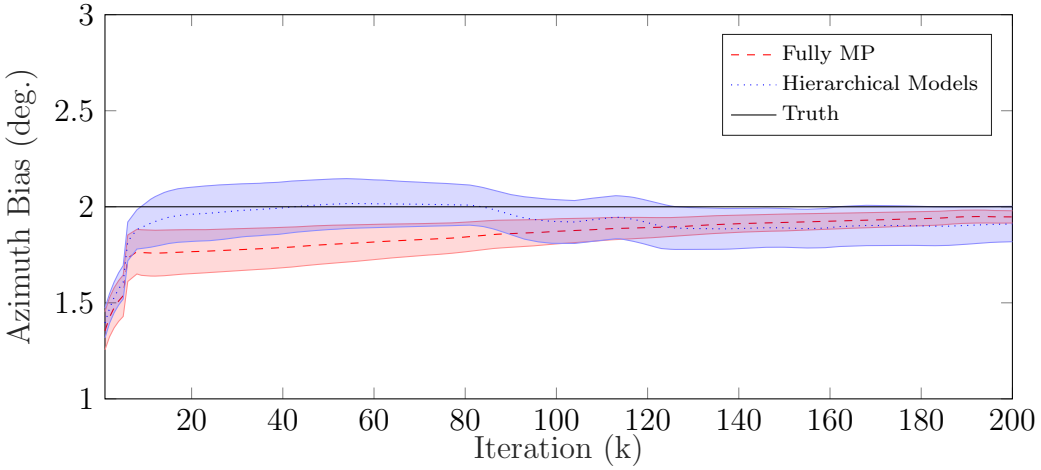
6.3.2 Experiment 2: Up to 16 Sensors, 100 Hypotheses

Following on from the results in Experiment 1, which showed that using beyond 100 parameter hypotheses gave no further performance gain in terms of both registration and fusion accuracy, the second experiment will test the scalability in terms of the number of sensors that are present in the region. The number of parameter hypotheses will now be fixed to 100, in order to estimate the relative biases between the reference sensor, and each of the other sensors providing measurements.

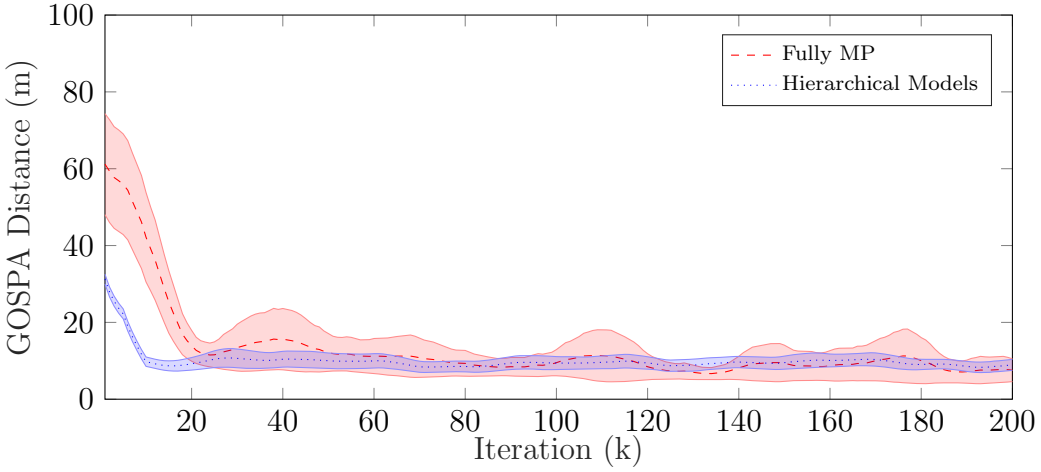
From this second set of results, it can again be seen in Figure 6.6 that there is no discernible, or statistical difference between the two methods in terms of estimation accuracy, both in the GOSPA results for tracking accuracy, and the RMSE results for sensor registration. By taking the average across the points plotted in Figure 6.6, the tabulated results in Table 6.2 can be found. It is positive to see that by adding more sensors to the network, there is no decrease in estimation accuracy, both in terms of the parameters and the target tracks. The key result is again in the timing



(a) Estimated range bias

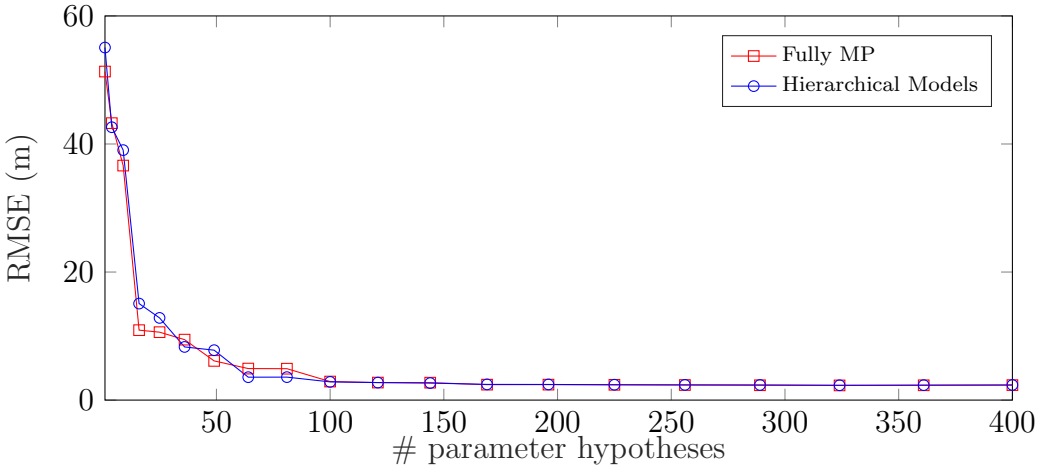


(b) Estimated azimuth bias

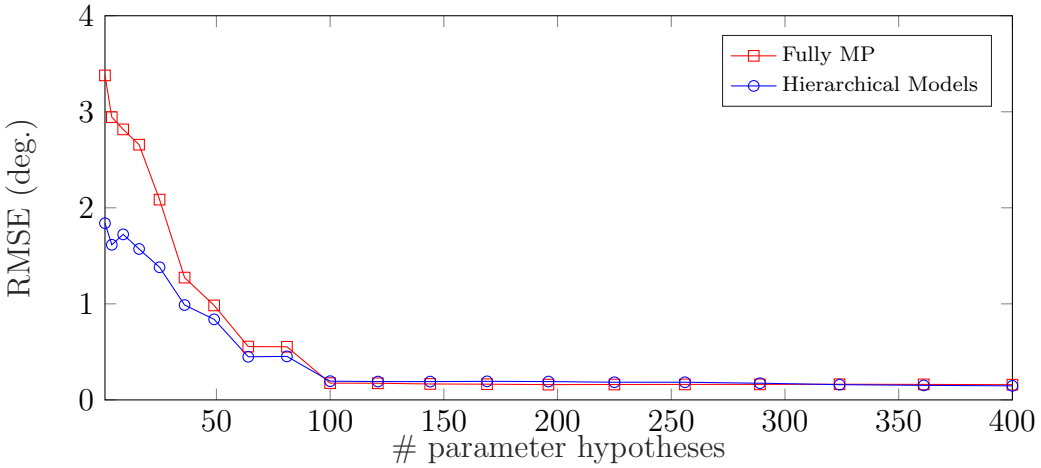


(c) GOSPA distance

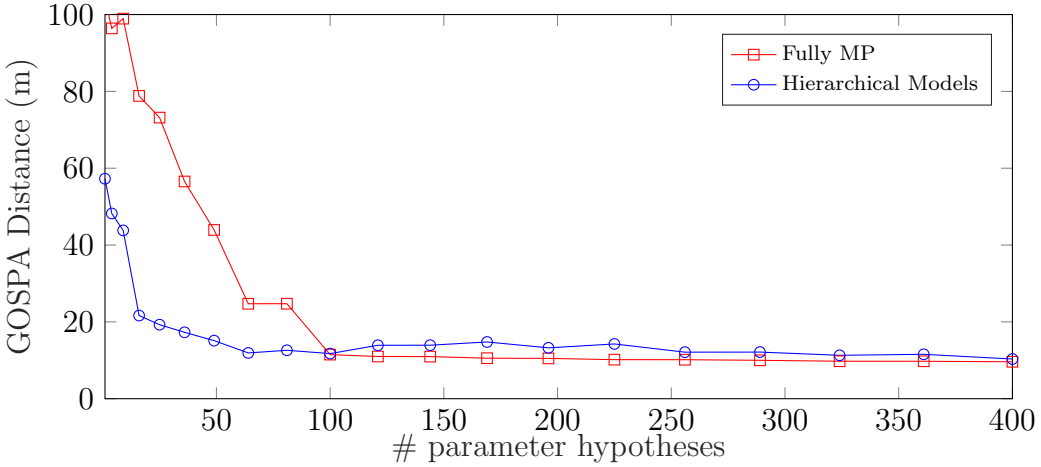
Figure 6.3: Parameter estimation and tracking accuracy, two sensors, $p_{\mathrm{d}} = 0.99$, 100 hypotheses



(a) Range RMSE



(b) Azimuth RMSE



(c) GOSPA Distance

Figure 6.4: Estimation accuracy over number of registration parameter hypotheses.

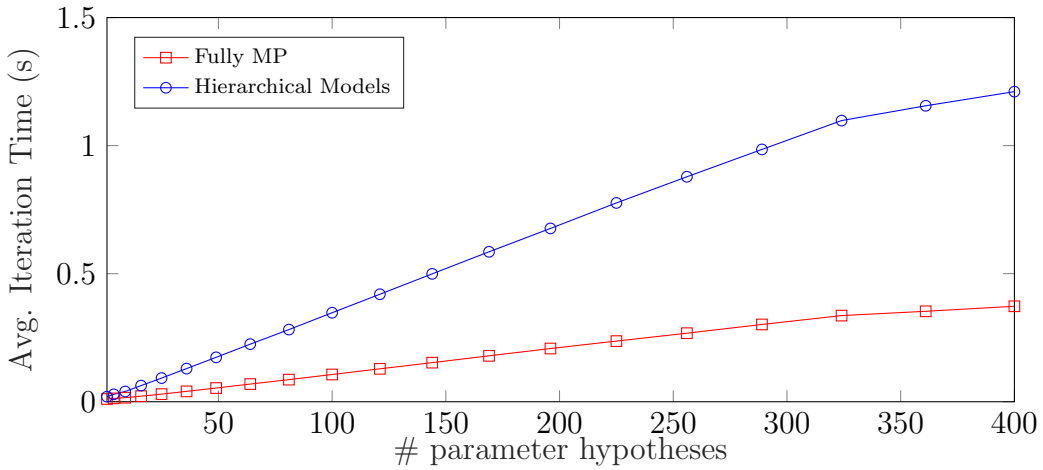


Figure 6.5: Execution time over number of registration parameter hypotheses.

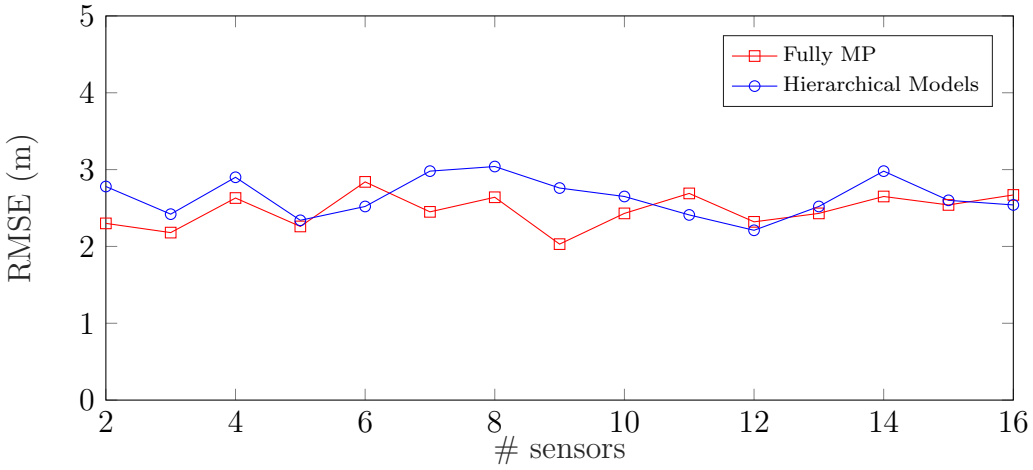
Table 6.2: Average Results for Experiment 2

Average Range RMSE	Fully MP	2.51 m
	Hierarchical Models	2.64 m
Average Azimuth RMSE	Fully MP	0.19°
	Hierarchical Models	0.21°
Average GOSPA Distance	Fully MP	9.58 m
	Hierarchical Models	9.57 m

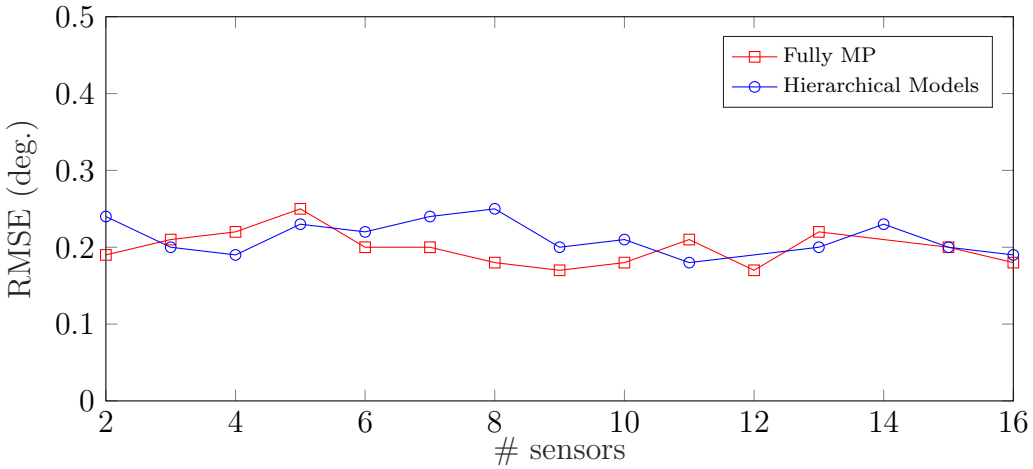
analysis of the two methods in Figure 6.7; the fully-MP approach is on average 92% faster at achieving a very similar result to that of the hierarchical model.

6.4 Conclusions

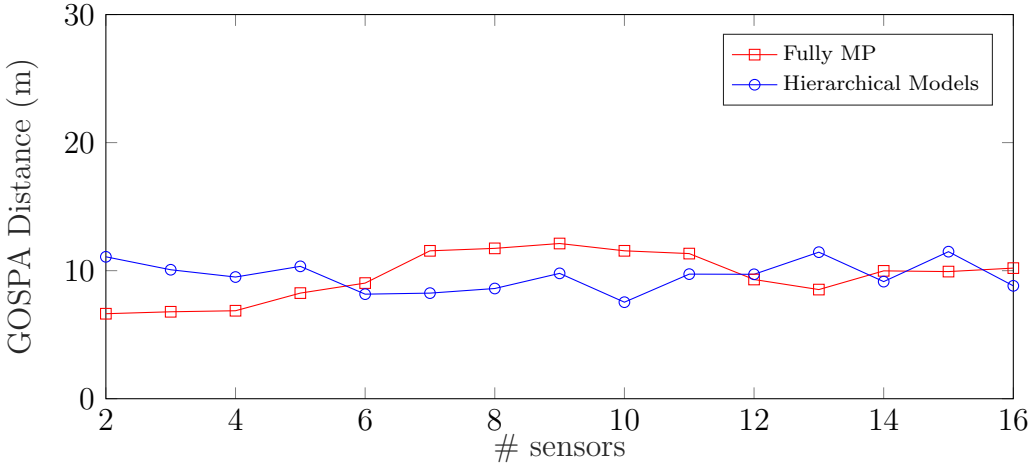
This chapter has shown that extending factor graphs to include parameter estimation is a useful and very efficient tool for resolving the joint registration and fusion problem. The results have shown that the advancements made to the algorithm that allow for continuous variables to be estimated, rather than just from discrete sets, have been successful, and provided a very similar outcome to the two previous chapters in terms of estimation accuracy. From the scalability tests that have been presented here, the average speed-up from using a fully-MP framework, rather than the hierarchical models developed earlier is approximately 158%. This large speed-up makes MP algorithms much more attractive for deployment in real systems, especially given that there is no decrease in accuracy.



(a) Range RMSE



(b) Azimuth RMSE



(c) GOSPA Distance

Figure 6.6: Estimation accuracy over number of sensors.

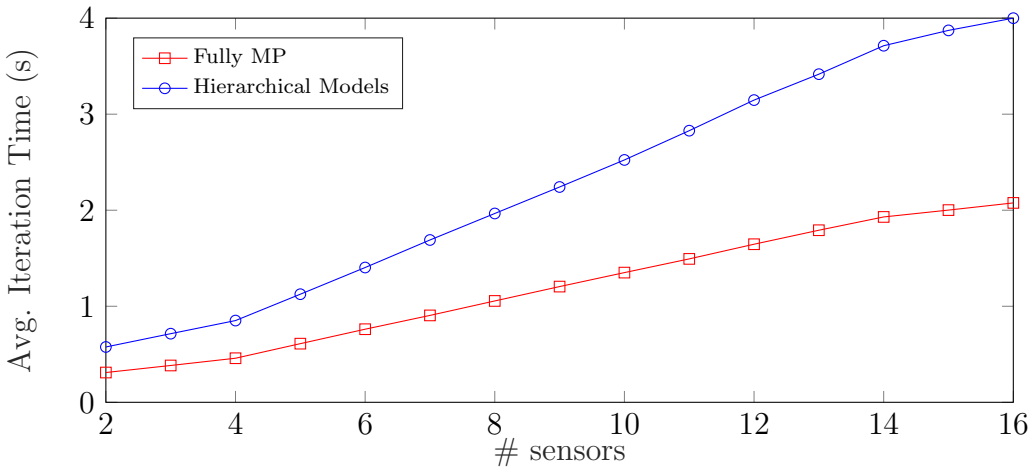


Figure 6.7: Execution time over number of sensors.

Chapter 7

Discussion and Conclusion

7.1 Summary

The main focus of this thesis was to investigate and develop scalable Multi-Target Tracking (MTT) and sensor fusion methods, which also incorporate sensor registration estimation in a joint manner. The work presented has been driven from an application perspective, and has taken problems that we may face in practice in to account, such as:

Limited resources being available on platforms, and hence the requirement for efficient and/or parallelisable algorithms.

Dynamic, time-varying biases that require a continual, online, calibration scheme.

Asynchronous sensors where there is no fixed time of arrival for the next set of sensor measurements, nor a guarantee which sensor will provide these measurements.

This thesis has shown that extending, and further developing, the *single-cluster method* to new applications of both sensor registration and fusion is an appropriate solution to these problems. It has been shown that this hierarchical approach is flexible, as a novel extension to Message Passing (MP) algorithms for MTT was successfully applied to this framework. This included the derivation of a suitable likelihood function to link the sensor registration and MTT processes together. The results presented on both the multistatic radar scenario and the heterogeneous case have highlighted the great importance of having accurate sensor registration knowledge, before attempting to perform any sort of sensor fusion. If the registration is left uncorrected, and biases remain in the measurements, the performance of the fusion algorithm is dramatically reduced. The methods that have been developed in this thesis have shown that it is possible to estimate appropriate sensor registration parameters jointly with the states of multiple dynamic targets in an efficient manner.

The *single-cluster method* framework presented in Chapter 4 that included the point process-based MTT algorithms, gave a suitable and robust technique that was able to accurately estimate the targets and parameters. However, the grid-based method was overly restrictive and attempting to scale the algorithm in that current setup would have resulted in an exponential growth in the execution time. In Chapter 5, a number of changes and further developments were made in order to gain more accurate estimation in both the sensor registration and the target tracking. The flexibility of the *single-cluster method* was proven as it was shown to be possible that alternative vector-based methods such as MP algorithms could be inserted into the framework. A suitable Multi-Object Likelihood (MOL) was derived in order to link the MTT with the newly-integrated Sequential Monte-Carlo (SMC) with resampling method that represented the sensor registration estimation process. Finally in Chapter 6, the scalability of the *single-cluster method* was put to the test, in comparison with a very recent development in the literature; a fully-MP framework capable of estimating discrete variables alongside target states. The final set of results have shown that in terms of estimation accuracy, both the *single-cluster method*, and the fully-MP framework, are very similar; however in terms of execution time, the fully-MP method can reach this accuracy in a much faster time. The scalability of the two methods are very similar; however the "constant" part is much less for the fully-MP method.

7.2 Future Directions

As a result of the work presented in this thesis, there are a number of interesting research routes that could be followed to extend this work. These include more complex and larger scenarios, considering alternative approximations in the formulation, and algorithmic design changes that could further improve performance of the proposed methods.

7.2.1 Dynamic Platforms

The simulated scenarios that have been presented in this thesis have all contained static sensors/platforms. The next steps on this thread could relate to the modelling and simulation of dynamic platforms, which, in turn, may contain dynamic biases such as Global Positioning System (GPS) drift through platform vibrations for example. For airborne platforms, there may also be issues with measurements coming from an Inertial Measurement Unit (IMU) as gyroscope readings could also contain bias if incorrectly calibrated before take-off.

Some related work has been carried out in different application areas, such as estimating the drift of a microscopy stage [147], correcting drift in telescope data [148] and also in aerial video registration [149]. These works however only consider

single sensor and single platform scenarios; ideas from these references could be adapted to suit dynamic situations.

7.2.2 Fusion Architectures

As discussed in Chapter 3, there are a number of different architectures in which sensor fusion can be deployed. However, the implementations have so far only focused on the centralised fusion framework, where the Fusion Centre (FC) has access to all of the raw measurement data that is output from each of the sensors after some pre-processing. If we were to consider a fully-distributed fusion architecture with track-to-track fusion instead, it would be of interest to see if a similar hierarchical model concept could directly be applied, or if it would require some adaptation.

Another interesting experiment would involve the hybrid fusion framework in a 2-platform case, where centralised fusion could be performed on each platform, then followed by distributed fusion between the platforms. This could raise a number of questions such as where should registration estimation be performed, as part of the centralised fusion, the distributed fusion, or even both?

7.2.3 Resampling Strategies

As described in [65], there are many resampling strategies available that could have been implemented as a part of the parent process in the hierarchical model, and for resampling the particles in the Belief Propagation (BP) algorithm. At the moment, the implementations use systematic resampling. For this work, and to further save on computational effort, it would be of interest to see if other resampling strategies would be more efficient. One idea may be to include a method that allows for sample sizes to be adapted over time, such as the work carried out in [150]. For example, once we become more confident in the estimation of the registration parameters and the variance of the particles reduces, we could begin to reduce the sample size in order to save further on resources. We could also arguably "switch off" the registration estimation completely if the uncertainty is sufficiently small; the embedded MTT algorithm of choice with the estimated parameters could be run on its own. Depending on the situation, it would be trivial to reintroduce the registration estimation if necessary.

7.2.4 BP Approximations

There are a plethora of techniques available in the BP literature that could be used in place of the particle BP algorithm used here. Other techniques use different approximations and have alternative representations of messages and beliefs. It would be interesting to use Gaussian Mixture BP (an alternative to the Extended Kalman Filter (EKF)-Probability Hypothesis Density (PHD) filter [91] used in Chapter 4),

analytical linearisation BP (similar to an EKF), or sigma-point BP (similar to an Unscented Kalman Filter (UKF) [59]) and perform a large comparison of these techniques to see which is most appropriate for MTT and fusion applications. It may also be possible to adapt the algorithm to other non-parametric techniques such as Kernel BP, where messages are represented as functions in Hilbert spaces.

7.2.5 Algorithm Programming/Parallelisation

The single-cluster method is potentially parallelisable, in that each of the particles used in the parent process is independent of one another while performing the filtering step. Previous work has been carried out in [108] to program the single-cluster method on to a Graphics Processing Unit (GPU), resulting in a large performance boost. This thesis only shows results for parallelisation on a Central Processing Unit (CPU) using four cores, which would be faster than serial processing. It would be interesting to see if the proposed hierarchical methods could be implemented on a GPU and test to see if any further speed-up was possible. If any speed-up was found, the hierarchical model results shown in Figure 6.5 and Figure 6.7 may be much closer to that of the fully-MP method.

When first attempting to replicate the MTT method that was presented in [22, 23], which in turn led to the formulation of the method described in Chapter 5, it was not possible to recreate both the timing and accuracy performance benchmarks given in the original papers. This highlights the importance of having an open repository of code for these types of algorithm, so that a fair, unbiased and rigorous comparison of techniques is possible. The current intention is to convert the methods presented in this thesis in to the Stone Soup repository [151–153] such that other tracking and fusion practitioners can access the code easily.

7.2.6 Estimation of System Model Parameters

The registration problem presented here has been focused on distance and orientation biases; however there are a number of other parameters that it would be useful to estimate if possible. For example, it could be very useful to try and estimate network parameters such as the time-varying latency between sensors and nodes, so that the Out-of-Sequence Measurement (OOSM) problems highlighted in Chapter 3 can be mitigated in an online, real-time manner.

Many of the parameters used in MTT are often set at the beginning of the scenario, and then kept fixed over the duration. In highly-dynamic scenarios (e.g. when performing maritime surveillance over rough seas, or in a dense urban environment) parameters such as the average false alarm rate, or the probability of detection may vary by large amounts over time. Being able to estimate and tune the false alarm rate in an online adaptive manner could result in more accurate tracks, and less

false tracks for example. It may also be possible to estimate other entities that are important for the tracking of targets. If more extensive data was available or simulated, we could potentially estimate target attributes such as target size.

7.2.7 Filter Overconfidence

Lastly, it has been shown in some pieces of literature [154, 155] that the EKF can sometimes become overconfident in its estimation of the uncertainty in a target's state, compared to the real uncertainty in reality. It has also been noted that other techniques based around the sigma-points method found in the UKF give more realistic and robust representations of the covariance/uncertainty. It would be interesting to see the performance of a UKF-PHD filter implementation to see how comparable the results are with the EKF-PHD filter implementation presented in this thesis.

Appendix A

Mathematical Tools

This appendix provides a summary of some mathematical operations and tools that can be used to manipulate point processes and Probability Generating Functionals (PGFLs) so that the PHD filter, Panjer filter, and their respective MOLs could be found. To begin with, let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space with a sample space Ω, $\sigma-$algebra $\mathcal{F}$ and probability measure $\mathbb{P}$. In the following sections, all of the random variables are defined on this probability space $(\Omega, \mathcal{F}, \mathbb{P})$.

A point process can be characterised by its Probability Generating Functional (PGFL)

$$G_\Phi(h) = \sum_{n\geq 0} \int \left[\prod_{i=1}^{n} h(x_i)\right] P_\Phi^{(n)}(x_1 \dots x_n) \mathrm{d}x_1 \dots \mathrm{d}x_n, \tag{A.1}$$

where $P_\Phi^{(n)}$ are the permutation-invariant probability measures on $\mathcal{X}^n$ for all $n \geq 0$, and h is a real-valued function. The permutation-invariance arises as it is assumed that permutations of the same sequence φ have the same probability. A PGFL is effectively a Probability Generating Function (PGF) that takes a function as its input, and a PGF is equivalent to a z-transform of a Probability Mass Function (PMF) [95,156]. Using PGFs allows for straight forward calculation of the moments of a point process; the nth moment around zero is the nth derivative of the Moment-Generating Function (MGF), evaluated at zero. The use of a PGFL in this case is very useful for mathematically representing a point process, as it gives a model of the number of objects by considering all of the cardinalities n in the summation in (A.1), and their stochastic properties (using the probabilities $P_\Phi^{(n)}$ for each fixed n).

A.1 Common PGFLs

Definition A.1.1 (Bernoulli process)**.** *Bernoulli processes are used to describe binary events; the event either happens with probability $p_\star$, or does not happen with probability $1 - p_\star$. With parameter $p_\star$, and spatial distribution $s_\star$, the PGFL of a*

Bernoulli process is

$$G_{Ber}(h) = (1 - p_\star) + p_\star \int h(x) s_\star(x) \mathrm{d}x. \tag{A.2}$$

A Bernoulli process can be used to describe the detection and survival of individual targets in the surveillance region as they only have two possible outcomes; either, the object no longer exists (or is not detected by the sensor) with probability $(1-p_\star)$*, or the object continues to exist (or is detected by the sensor) with probability* $p_\star$*, and will evolve according to the chosen spatial distribution* $s_\star(\cdot)$*.*

Definition A.1.2 (Poisson process)**.** *A Poisson process with parameter* λ*, and spatial distribution* $s_\star$ *is an independent and identically distributed (i.i.d.) cluster process with a spatial distribution* $s_\star$*, whose size is Poisson distributed with rate* λ*. The PGFL of a Poisson point process is [157]*

$$G_{Poi}(h) = \exp\left(\int \left[h(x) - 1 \right] \mu(x) \mathrm{d}x \right), \tag{A.3}$$

where the intensity of the process $\mu(\cdot) = \lambda s_\star(\cdot)$*. The compact form shown in (A.3) is found by substituting the PMF of a Poisson distribution in to (A.1), and then exploiting the series expansion of an exponential function to remove the infinite sum.*

Definition A.1.3 (Panjer process)**.** *A Panjer point process with parameters* α *and* β*, and a spatial distribution* $s_\star$ *is an i.i.d. cluster process with spatial distribution* $s_\star$*. The size of the described population is modelled with a Panjer distribution with parameters* α *and* β *[96]. The PGFL for the Panjer process is [97, 139]*

$$G_{Pan}(h) = \left(1 + \frac{1}{\beta} \int \left[1 - h(x) \right] s_\star(x) \mathrm{d}x \right)^{-\alpha}. \tag{A.4}$$

By taking the limit $\alpha \to \infty$, and keeping the ratio $\frac{\alpha}{\beta}$ constant, Equation (A.4) will reduce down to the Poisson PGFL in Equation (A.3) with $\lambda = \frac{\alpha}{\beta}$. By choosing negative values for both α and β according to the properties listed in Section 2.3.3, a binomial process can be obtained. Conversely, if positive values are chosen, a negative binomial process is found. This makes the Panjer process very versatile and allows for much more flexibility in modelling choices than the Poisson process. These Panjer use cases are presented in more detail in Section 2.3.3.

A.2 Working with PGFLs

In order to compute explicit formulae for the MOLs, we can use the chain differential, which will allow for a higher-order product and chain rule [158]. Note that other

common differential operators such as the Gâteaux and Fréchet differential could be exploited [159].

Definition A.2.1 (Chain differential [158, 159])**.** *Let* $(\eta_n : \mathcal{X} \to \mathbb{R}^+)_{n\in\mathbb{N}}$ *be a sequence of positive, bounded functions converging pointwise to a function* $\eta : \mathcal{X} \to \mathbb{R}^+$ *and let* $(\epsilon_n)_{n\in\mathbb{N}}$ *be a sequence of positive real values converging to* 0*. The chain differential of a functional* G *with respect to its functional argument* $h : \mathcal{X} \to \mathbb{R}^+$ *in the direction of* η *can be defined as*

$$\delta G(h;\eta) = \lim_{n\to\infty} \frac{G(h+\epsilon_n\eta_n) - G(h)}{\epsilon_n} \tag{A.5}$$

If this limit exists, it will be unique for any sequence $(\epsilon_n)_{n\in\mathbb{N}}$ *and* $(\eta_n : \mathcal{X} \to \mathbb{R}^+)_{n\in\mathbb{N}}$ *with the above properties.*

The chain differential gives both an n-fold product rule [157]

$$\delta^n(F\cdot G)(h;\eta_1,\ldots,\eta_n) = \sum_{\omega\subseteq\{1,\ldots,n\}} \delta^{|\omega|}F\left(h;(\eta_i)_{i\in\omega}\right)\delta^{|\bar{\omega}|}G\left(h;(\eta_j)_{j\in\bar{\omega}}\right) \tag{A.6}$$

with $\bar{\omega}$ being a set complement $\{1,\ldots,n\}\backslash\omega$, and an n-fold chain rule (Faà di Bruno's formula for chain differentials) [158]

$$\delta^n(F\circ G)(h;\eta_1,\ldots,\eta_n) = \sum_{\pi\in\prod_n} \delta^{|\pi|}F\left(G(h);\left(\delta^{|\omega|}G(h;(\eta_i)_{i\in\omega})\right)_{\omega\in\pi}\right). \tag{A.7}$$

Faà di Bruno's formula provides a general form of the chain rule for higher-order derivatives, which in turn provides a useful tool for finding the higher-order moments of a distribution.

By differentiating a PGFL, the statistical moments of a point process can be obtained, and with those, more properties can be found. Two properties that are of interest are

1. Calculating the expected value:

$$\mathbb{E}[G_\Phi] = \delta G_\Phi(h;\eta)|_{h=1} \tag{A.8}$$

2. Extracting the probability measure of having exactly n objects:

$$P_\Phi^{(n)}(x_1,\ldots,x_n) = \delta^n G_\Phi(h;\delta_{x_1},\ldots,\delta_{x_n})|_{h=0}, \tag{A.9}$$

where δ_x is the Dirac delta function, being non-zero at x only [160]. Equation (A.9) is important in deriving the MOLs for these filters, as we need the likelihood of obtaining exactly m measurements $\mathbf{z}_1,\ldots,\mathbf{z}_m$, given the current sensor registration configuration. Many problems that fall under the Bayesian filtering framework

cannot be fully described using a single point process; several processes are often combined in the prediction and update steps. Joint PGFLs can be written in a similar form to that of Equation (A.1), but instead with joint probabilities $P_{\Phi}^{(n,m)}(x_1, \ldots, x_n, \mathbf{z}_1, \ldots, \mathbf{z}_m)$. In general, this does not simplify, but there are two special cases of point process concatenation that are of interest, namely superposition and branching [95, 156].

1. Superposition: If the two point processes are independent of one another, their joint PGFL will decompose into a product of the form

$$G_{\Phi,\Psi}(h, g) = G_{\Phi}(h)G_{\Psi}(g). \tag{A.10}$$

   This case is used when adding the spontaneous clutter to the model, as the clutter is assumed to be independent of the target process.

2. Branching: If each point in the point process Φ creates a new point process Ψ, the resultant PGFL is a concatenation of the individual functionals

$$G_{\Phi,\Psi}(h, g) = G_{\Phi}(hG_{\Psi}(g|\cdot)). \tag{A.11}$$

   This type of structure is required to represent the detection process of the current targets. For example, each target is either detected, or not detected, which could be represented using a Bernoulli process (Equation (A.2)); the predicted target process could be represented with a Poisson process (Equation (A.3)).

By exploiting these cases, the general form of the PGFL that describes the joint measurement and target processes, dependent on the sensor registration configuration q, is of the form

$$G_J(g, h|q) = G_{pr}(hG_d(g|\cdot, q))G_c(g|q) \tag{A.12}$$

where $G_d(g|\cdot, q)$ characterises the Bernoulli detection process for a given target state x and registration configuration q, with detection probability $p_\star(x) = p_d(x|q)$ and single-object measurement association likelihood $s_\star(\cdot) = l(x|\cdot, q)$ such that

$$G_d(g|x, q) = 1 - p_d(x|q) + p_d(x|q) \int g(z)l(x|z, q)\mathrm{d}z \tag{A.13}$$

and where G_{pr} is the PGFL of the predicted target process, and G_c is the PGFL of the clutter process [112].

References

[1] M. Dana, "Registration: A Prerequisite for Multiple Sensor Tracking," in *Multitarget-Multisensor Tracking: Advanced Applications*, Y. Bar-Shalom, Ed. Norwood, MA: Artech House, 1990, pp. 1–392.

[2] P. Horridge and S. Maskell, "Real-Time Tracking Of Hundreds Of Targets With Efficient Exact JPDAF Implementation," in *9th International Conference on Information Fusion (FUSION)*. Florence, Italy: IEEE, 2006.

[3] D. Cormack and D. Clark, "Tracking Small UAVs Using a Bernoulli Filter," in *2016 Sensor Signal Processing for Defence, SSPD 2016*, 2016.

[4] M. Taylor, "Twelve protesters arrested over Heathrow drone threat," 2019. [Online]. Available: https://www.theguardian.com/uk-news/2019/sep/13/heathrow-protests-two-held-near-airport-as-activists-threaten-drone-disruption

[5] S. Calder, "GATWICK DRONE DISRUPTION COST OVER £50M," 2019. [Online]. Available: https://www.independent.co.uk/travel/news-and-advice/gatwick-drone-airport-cost-easyjet-runway-security-passenger-cancellation-a8739841.html

[6] "Gatwick Airport: Drones ground flights," 2018. [Online]. Available: https://www.bbc.co.uk/news/uk-england-sussex-46623754

[7] J. Hall and P. Allen, "New terror alert for France after 'at least' five drones are spotted flying above Paris landmarks including Eiffel Tower and the US embassy," 2015. [Online]. Available: https://www.dailymail.co.uk/news/article-2966482/New-terror-alert-France-five-drones-spotted-flying-Paris-landmarks-including-Eiffel-Tower-embassy.html

[8] N. Ames, "Serbia v Albania: Drones, flags and violence in abandoned match," 2014. [Online]. Available: https://www.bbc.co.uk/sport/football/29624259

[9] "US Open: New York teacher arrested after drone crashes into stands," 2015. [Online]. Available: https://www.bbc.co.uk/news/world-us-canada-34155773

[10] "Warning over drones use by terrorists," 2016. [Online]. Available: https://www.bbc.co.uk/news/technology-35280402

[11] W. Koch, *Tracking and Sensor Data Fusion: Methodological Framework and Selected Applications*, 1st ed. Berlin: Springer Verlag, 2014.

[12] E. Taghavi, R. Tharmarasa, T. Kirubarajan, and M. Mcdonald, "Multisensor-multitarget bearing-only sensor registration," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 52, no. 4, pp. 1654–1666, 2016.

[13] E. Taghavi, R. Tharmarasa, T. Kirubarajan, Y. Bar-Shalom, and M. Mcdonald, "A Practical Bias Estimation Algorithm for Multisensor-Multitarget Tracking," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 52, no. 1, pp. 2 – 19, 2016.

[14] Z. Li, S. Chen, H. Leung, and E. Bosse, "Joint Data Association, Registration, and Fusion using EM-KF," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 46, no. 2, pp. 496 – 507, 2010.

[15] D. Huang, H. Leung, and E. Bosse, "A Pseudo-Measurement Approach to Simultaneous Registration and Track Fusion," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 48, no. 3, pp. 2315 – 2331, 2012.

[16] 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, 2005.

[17] ——, "Exact Multisensor Dynamic Bias Estimation with Local Tracks," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 40, no. 2, pp. 576 – 590, 2004.

[18] Y. Bar-Shalom, *Multitarget-Multisensor Tracking: Applications and Advances Volume III*, 1st ed., W. D. Blair, Ed. Norwood, MA: Artech House, 2000.

[19] B.-N. Vo, M. Mallick, Y. Bar-Shalom, S. Coraluppi, R. Osborne, R. Mahler, and B.-T. Vo, "Multitarget Tracking," in *Wiley Encyclopedia of Electrical and Electronics Engineering*. American Cancer Society, 2015, pp. 1–15.

[20] S. Maresca, P. Braca, J. Horstmann, and R. Grasso, "Maritime Surveillance Using Multiple High-Frequency Surface-Wave Radars," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 8, pp. 5056 – 5071, 2014.

[21] Y. Bar-Shalom, P. K. Willett, and X. Tian, *Tracking and Data Fusion: A Handbook of Algorithms*. Storrs, CT, USA: YBS Publishing, 2011.

[22] F. Meyer, P. Braca, P. Willett, and F. Hlawatsch, "A Scalable Algorithm for Tracking an Unknown Number of Targets Using Multiple Sensors," *IEEE Transactions on Signal Processing*, vol. 65, no. 13, pp. 3478–3493, 2017.

[23] F. Meyer, T. Kropfreiter, J. L. Williams, R. A. Lau, F. Hlawatsch, P. Braca, and M. Z. Win, "Message Passing Algorithms for Scalable Multitarget Tracking," *Proceedings of the IEEE*, vol. 106, no. 2, pp. 221–259, 2018.

[24] M. Mallick, K.-C. Chang, S. Arulampalam, and Y. Yan, "Heterogeneous Track-to-Track Fusion in 3D Using IRST Sensor and Air MTI Radar," *IEEE Transactions on Aerospace and Electronic Systems*, 2019.

[25] J. Yan, H. Liu, W. Pu, B. Jiu, Z. Liu, and Z. Bao, "Benefit Analysis of Data Fusion for Target Tracking in Multiple Radar System," *IEEE Sensors Journal*, vol. 16, no. 16, pp. 6359 – 6366, 2016.

[26] S. Wu, S. Decker, P. Chang, T. Camus, and J. Eledath, "Collision Sensing by Stereo Vision and Radar Sensor Fusion," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 4, pp. 606 – 614, 2009.

[27] G. Mirzaei, M. M. Jamali, J. Ross, P. V. Gorsevski, and V. P. Bingman, "Data Fusion of Acoustics, Infrared and Marine Radar for Avian Study," *IEEE Sensors Journal*, vol. 15, no. 11, pp. 6625–6632, 2015.

[28] C. Huazhi and R. Jian, "A Multitarget Tracking Algorithm based on Radar and Infrared Sensor Data Fusion," in *2011 IEEE 3rd International Conference on Communication Software and Networks (ICCSN)*. Xi'an, China: IEEE, 2011, pp. 367–371.

[29] H. Mitchell, *Multi-Sensor Data Fusion: An Introduction*, 1st ed. Berlin, Germany: Springer-Verlag, 2007.

[30] W. Elmenreich, "An Introduction to Sensor Fusion," Vienna Institute of Technology, Austria, Vienna, Austria, Tech. Rep., 2002.

[31] B. Ristic, D. Clark, and N. Gordon, "Calibration of Multi-Target Tracking Algorithms Using Non-Cooperative Targets," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 3, pp. 390–398, 2013.

[32] B. Siciliano and O. Khatib, *Springer Handbook of Robotics*, illus. ed., B. Siciliano and O. Khatib, Eds. Springer Science and Business Media, 2008.

[33] G. Dudek and M. Jenkin, *Computational Principles of Mobile Robotics*. New York, USA: Cambridge University Press, 2000.

[34] J. Vermaak, S. Maskell, and M. Briers, "Online Sensor Registration," in *2005 IEEE Aerospace Conference*. Big Sky, Montana: IEEE, 2005, pp. 2117–2125.

[35] D. Cormack, I. Schlangen, J. R. Hopgood, and D. E. Clark, "Joint Registration and Fusion of an Infra-Red Camera and Scanning Radar in a Maritime Context," *IEEE Transactions on Aerospace and Electronic Systems*, 2019.

[36] G. Stimson, *Stimson's Introduction to Airborne Radar*, 3rd ed., H. Griffiths, C. Baker, and D. Adamy, Eds. SciTech Publishing, 2014.

[37] H. Karniely and H. Siegelmann, "Sensor registration using neural networks," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 36, no. 1, pp. 85–101, 2000.

[38] N. Schneider, F. Piewak, C. Stiller, and U. Franke, "RegNet: Multimodal sensor registration using deep neural networks," in *2017 IEEE Intelligent Vehicles Symposium*. Los Angeles, CA: IEEE, 2017, pp. 1803–1810.

[39] G. Marcus, "Deep Learning: A Critical Appraisal," *ArXiv e-prints, ArXiv: 1801.00631*, pp. 1–27, 2018. [Online]. Available: https://arxiv.org/abs/1801.00631

[40] P. Domingos, *The Master Algorithm: How the Quest for the Ultimate Learning Machine Will Remake Our World*, 1st ed. London, UK: Allen Lane, 2015.

[41] M. Skolnik, *Introduction to Radar Systems*, 3rd ed. New York: McGraw-Hill, 2001.

[42] S. M. Sherman and D. K. Barton, *Monopulse Principles and Techniques*, 2nd ed. Artech House, 2011.

[43] V. S. Chernyak, *Fundamentals of Multisite Radar Systems: Multistatic Radars and Multiradar Systems*, 1st ed. CRC Press, 1998.

[44] S. R. Doughty, "Development and performance evaluation of a multistatic radar system," Ph.D. dissertation, University College London, 2008.

[45] D. Tahmoush, "Detection of Small UAV Helicopters Using Micro-Doppler," in *SPIE 9077, Radar Sensor Technology XVIII*. Baltimore, Maryland: SPIE, 2014, p. 6.

[46] A. R. Persico, C. Clemente, L. Pallotta, A. De Maio, and J. Soraghan, "Micro-Doppler classification of ballistic threats using Krawtchouk moments," in *2016 IEEE Radar Conference (RadarConf)*. Philadelphia, PA: IEEE, 5 2016, pp. 121–126. [Online]. Available: http://ieeexplore.ieee.org/document/7485086/

[47] S. Watts, "Radar Sea Clutter: Recent Progress and Future Challenges," in *2008 International Conference on Radar*. Adelaide, Australia: IEEE, 2008, pp. 10–16.

[48] ——, "Modeling and Simulation of Coherent Sea Clutter," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 48, no. 4, pp. 3303–3317, 2012.

[49] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman Filter: Particle Filters for Tracking Applications*, 1st ed. Boston, MA: Artech House, 2004.

[50] G. Soldi, F. Meyer, P. Braca, and F. Hlawatsch, "Self-Tuning Algorithms for Multisensor-Multitarget Tracking Using Belief Propagation," *IEEE Transactions on Signal Processing*, vol. 67, no. 15, pp. 3922 – 3937, 2019.

[51] X. Rong-Li and V. P. Jilkov, "Survey of maneuvering target tracking. Part V. Multiple-model methods," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 41, no. 4, pp. 1255 – 1321, 2005.

[52] X. Rong-Li and V. Jilkov, "Survey of maneuvering target tracking. Part I. Dynamic models," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 4, pp. 1333 – 1364, 2003.

[53] G. A. Watson and W. D. Blair, "IMM algorithm for tracking targets that maneuver through coordinated turns," in *Proc. SPIE 1698, Signal and Data Processing of Small Targets 1992*. Orlando, FL: SPIE, 1992, pp. 236 – 247.

[54] R. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *ASME Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[55] A. H. Jazwinski, *Stochastic Processes and Filtering Theory*. London: Academic Press, 1970.

[56] G. Haag, "Derivation of the Chapman-Kolmogorov Equation and the Master Equation," in *Modelling with the Master Equation: Solution Methods and Applications in Social and Natural Sciences*, 1st ed. Cham: Springer, 2017, ch. 3, pp. 39 – 61.

[57] Y. Bar-Shalom, X. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation: Theory, Algorithms and Software*, 1st ed. New York, NY: John Wiley & Sons, 2001.

[58] B. A. McElhoe, "An Assessment of the Navigation and Course Corrections for a Manned Flyby of Mars or Venus," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-2, no. 4, pp. 613–623, 1966.

[59] S. Julier, J. Uhlmann, and H. Durrant-White, "A New Method for Nonlinear Transformation of Means and Covariances in Filters and Estimators," *IEEE Transactions on Automatic Control*, vol. 45, no. 3, pp. 477–482, 2000.

[60] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401 – 422, 2004.

[61] I. Arasaratnam and S. Haykin, "Cubature Kalman Filters," *IEEE Transactions on Automatic Control*, vol. 54, no. 6, pp. 1254 – 1269, 2009.

[62] A. Doucet, N. de Freitas, and N. Gordon, *Sequential Monte Carlo Methods in Practice*. New York, NY: Springer-Verlag, 2001.

[63] M. S. 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, 2002.

[64] N. Gordon, D. Salmond, and A. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *IEE Proceedings F - Radar and Signal Processing*, vol. 140, no. 2, pp. 107 – 113, 1993.

[65] T. Li, M. Bolic, and P. M. Djuric, "Resampling Methods for Particle Filtering: Classification, Implementation and Strategies," *IEEE Signal Processing Magazine*, vol. 32, no. 3, pp. 70–86, 2015.

[66] B. Ristic, B.-T. Vo, B.-N. Vo, and A. Farina, "A Tutorial on Bernoulli Filters: Theory, Implementation and Applications," *IEEE Transactions on Signal Processing*, vol. 61, no. 13, pp. 3406 – 3430, 2013.

[67] J. Houssineau and D. Laneuville, "PHD filter with diffuse spatial prior on the birth process with applications to GM-PHD filter," in *13th International Conference on Information Fusion*. Edinburgh, UK: IEEE, 2010, pp. 1 – 8.

[68] K. Granstrom, M. Baum, and S. Reuter, "Extended Object Tracking: Introduction, Overview, and Applications," *Journal of Advances in Information Fusion*, vol. 12, no. 2, pp. 139 – 174, 2017.

[69] H. A. Blom and E. A. Bloem, "Bayesian tracking of two possibly unresolved maneuvering targets," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 43, no. 2, pp. 612 – 627, 2007.

[70] H. W. Kuhn, "The Hungarian Method for the Assignment Problem," *Naval Research Logistics Quarterly*, vol. 2, pp. 83–97, 1955.

[71] J. Munkres, "Algorithms for Assignment and Transportation Problems," *Journal of the Society for Industrial and Applied Mathematics*, vol. 5, no. 1, pp. 32 – 38, 1957.

[72] F. Bourgeois and J.-C. Lassalle, "An Extension of the Munkres Algorithm for the Assignment Problem to Rectangular Matrices," *Communications of the ACM*, vol. 14, no. 12, pp. 802 – 804, 12 1971.

[73] D. Bertsekas, "A New Algorithm for the Assignment Problem," *Mathematical Programming*, vol. 21, no. 1, pp. 152–171, 1981.

[74] D. Bertsekas and D. Castanon, "A forward/reverse auction algorithm for asymmetric assignment problems," *Computational Optimization and Applications*, vol. 1, pp. 277 – 297, 1992.

[75] M. Levedahl, "Performance comparison of 2D assignment algorithms for assigning truth objects to measured tracks," in *SPIE 4048, Signal and Data Processing of Small Targets 2000*. Orlando, FL: SPIE, 2000, pp. 380 – 390.

[76] D. P. Bertsekas, "Auction Algorithms for Network Flow Problems: A Tutorial Introduction," *Computational Optimization and Applications*, vol. 1, pp. 7 – 66, 1992.

[77] J. Williams and R. Lau, "Approximate evaluation of marginal association probabilities with belief propagation," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 50, no. 4, pp. 2942 – 2959, 2014.

[78] F. Kschischang, B. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498 – 519, 2001.

[79] J. L. Williams and R. A. Lau, "Multiple Scan Data Association by Convex Variational Inference," *IEEE Transactions on Signal Processing*, vol. 66, no. 8, pp. 2112 – 2127, 2018.

[80] J. Houssineau and D. E. Clark, "On a representation of partially-distinguishable populations," *A Journal of Theoretical and Applied Statistics*, vol. 54, no. 1, pp. 23 – 45, 2020.

[81] S. Blackman and R. Popoli, *Design and Analysis of Modern Tracking Systems*. Artech House, 1999.

[82] T. Fortmann, Y. Bar-Shalom, and M. Scheffe, "Multi-Target Tracking using Joint Probabilistic Data Association," in *19th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes*. Albuquerque, NM, USA: IEEE, 1980, pp. 807–812.

[83] ——, "Sonar Tracking of Multiple Targets using Joint Probabilistic Data Association," *IEEE Journal of Oceanic Engineering*, vol. 8, no. 3, pp. 173–184, 1983.

[84] S. Maskell, M. Briers, and R. Wright, "Fast mutual exclusion," in *Proc. SPIE 5428, Signal and Data Processing of Small Targets*. Orlando, FL: SPIE, 2004.

[85] J. Roecker, "A class of near optimal JPDA algorithms," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 30, no. 2, pp. 504–510, 1994.

[86] K. Romeo, D. Crouse, Y. Bar-Shalom, and P. Willett, "The JPDAF in practical systems: approximations," in *Proc. SPIE 7698, Signal and Data Processing of Small Targets*. Orlando, FL.: SPIE, 2010.

[87] D. Musicki, R. Evans, and S. Stankovic, "Integrated probabilistic data association," *IEEE Transactions on Automatic Control*, vol. 39, no. 6, pp. 1237 – 1241, 1994.

[88] D. B. Reid, "An Algorithm for Tracking Multiple Targets," *IEEE Transactions on Automatic Control*, vol. 24, no. 6, pp. 843–854, 1979.

[89] S. Blackman, "Multiple Hypothesis Tracking for Multiple Target Tracking," *IEEE Aerospace and Electronics Systems Magazine*, vol. 19, no. 1, pp. 5–18, 2004.

[90] T. Kurien, "Issues in the Design of Practical Multitarget Tracking Algorithms," in *Multitarget-Multisensor Tracking: Advanced Applications*, Y. Bar-Shalom, Ed. Norwood, MA: Artech House, 1990, ch. 3, pp. 43 – 83.

[91] B.-N. Vo and W.-K. Ma, "The Gaussian Mixture Probability Hypothesis Density Filter," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4091–4104, 2006.

[92] R. Mahler, "Multitarget Bayes filtering via first-order multitarget moments," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 4, pp. 1152–1178, 2003.

[93] ——, *Statistical Multisource-Multitarget Information Fusion*. Norwood, MA: Artech House, 2007.

[94] ——, "PHD Filters of Higher Order in Target Number," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 43, no. 4, pp. 1523–1543, 2007.

[95] I. Schlangen, "Multi-object filtering with second-order moment statistics," Ph.D. dissertation, Heriot-Watt University, 2017.

[96] M. Fackler, "Panjer class united – one formula for the Poisson, binomial and negative binomial distribution," in *39th International ASTIN Colloquium*, Helsinki, Finland, 2009, pp. 1 – 9.

[97] I. Schlangen, E. D. Delande, J. Houssineau, and D. E. Clark, "A Second-Order PHD Filter With Mean and Variance in Target Number," *IEEE Transactions on Signal Processing*, vol. 66, no. 1, pp. 48–63, 1 2018.

[98] S. Nagappa, D. E. Clark, and R. Mahler, "Incorporating Track Uncertainty into the OSPA Metric," in *2011 Proceedings of the 14th International Conference on Information Fusion (FUSION)*. Chicago, IL: IEEE, 2011, p. 8.

[99] B.-N. Vo, S. Singh, and A. Doucet, "Sequential Monte Carlo methods for multitarget filtering with random finite sets," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 41, no. 4, pp. 1224 – 1245, 2005.

[100] K. Gilholm and D. Salmond, "Spatial distribution model for tracking extended objects," *IEE Proceedings - Radar, Sonar and Navigation*, vol. 152, no. 5, 2005.

[101] B.-T. Vo, B.-N. Vo, and A. Cantoni, "Analytic Implementations of the Cardinalized Probability Hypothesis Density Filter," *IEEE Transactions on Signal Processing*, vol. 55, no. 7, pp. 3553–3567, 7 2007.

[102] K. D. Ward, R. J. Tough, and S. Watts, *Sea Clutter: Scattering, the K Distribution and Radar Performance*, 1st ed. London, U.K.: The Institution of Engineering and Technology, 2013.

[103] D. Cormack and J. R. Hopgood, "Sensor Registration and Tracking from Heterogeneous Sensors with Belief Propagation," in *22nd International Conference on Information Fusion (FUSION)*. Ottawa, CA: IEEE, 2019.

[104] B.-N. Vo, B.-T. Vo, and D. Phung, "Labeled Random Finite Sets and the Bayes Multi-Target Tracking Filter," *IEEE Transactions on Signal Processing*, vol. 62, no. 24, pp. 6554 – 6567, 2014.

[105] B.-N. Vo, B.-T. Vo, and H. G. Hoang, "An Efficient Implementation of the Generalized Labeled Multi-Bernoulli Filter," *IEEE Transactions on Signal Processing*, vol. 65, no. 8, pp. 1975 – 1987, 2017.

[106] C. S. Lee, D. Clark, and J. Salvi, "SLAM with Dynamic Targets via Single-Cluster PHD Filtering," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 3, pp. 543–552, 2013.

[107] C. S. Lee, D. E. Clark, and J. Salvi, "SLAM with single cluster PHD filters," in *2012 IEEE International Conference on Robotics and Automation*. Saint Paul, MN: IEEE, 2012, pp. 2096 – 2101.

[108] C. S. Lee, J. Franco, J. Houssineau, and D. Clark, "Accelerating the Single Cluster PHD Filter with a GPU Implementation," in *2014 International Conference on Control, Automation and Information Sciences (ICCAIS)*. Gwangju, South Korea: IEEE, 2014, pp. 53–58.

[109] 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, 2011.

[110] A. Swain and D. Clark, "The single-group PHD filter: An analytic solution," in *2011 Proceedings of the 14th International Conference on Information Fusion (FUSION)*. Chicago, IL: IEEE, 2011.

[111] A. Swain, "Group and extended target tracking with the probability hypothesis density filter," Ph.D. dissertation, Heriot-Watt University, 2013.

[112] I. Schlangen, D. E. Clark, and E. D. Delande, "Single-cluster PHD filter methods for joint multi-object filtering and parameter estimation," *ArXiv e-prints, ArXiv:1705.05312*, 5 2017. [Online]. Available: http://arxiv.org/abs/1705.05312

[113] J. F. Steffensen, *Interpolation*, 2nd ed. Mineola, NY: Dover Publications, 2009.

[114] S. Nagappa and D. E. Clark, "On the ordering of the sensors in the iterated-corrector probability hypothesis density (PHD) filter," in *Proc. SPIE 8050, Signal Processing, Sensor Fusion, and Target Recognition XX, 80500M*. Orlando, FL: SPIE, 2011, pp. 1 – 6.

[115] F. Kschischang and B. Frey, "Iterative Decoding of Compound Codes by Probability Propagation in Graphical Models," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 2, pp. 219 – 230, 1998.

[116] B. Frey, *Graphical Models for Machine Learning and Digital Communication*. Cambridge, MA: MIT Press, 1998.

[117] J. Pearl, "Fusion, propagation, and structuring in belief networks," *Artificial Intelligence*, vol. 29, no. 3, pp. 241–288, 1986.

[118] S. Lauritzen and D. Spiegelhalter, "Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems," *Journal of the Royal Statistical Society, Series B (Methodological)*, vol. 50, no. 2, pp. 157–224, 1988.

[119] G. Shafer, *Probabilistic Expert Systems*. Philadelphia, PA: Society for Industrial & Applied Mathematics, 1996.

[120] V. Lepar and P. Shenoy, "A Comparison of Lauritzen-Spiegelhalter, Hugin, and Shenoy-Shafer Architectures for Computing Marginals of Probability Distributions," in *Fourteenth Conference on Uncertainty in Artificial Intelligence*

(UAI1998). Madison, WI: Morgan Kaufmann Publishers Inc., 1998, pp. 328 – 337.

[121] M. Uney, B. Mulgrew, and D. E. Clark, "A Cooperative Approach to Sensor Localisation in Distributed Fusion Networks," *IEEE Transactions on Signal Processing*, vol. 64, no. 5, pp. 1187–1199, 2016.

[122] ——, "Latent Parameter Estimation in Fusion Networks Using Separable Likelihoods," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 4, no. 4, pp. 752 – 768, 2018.

[123] M. Uney, K. Copsey, S. Page, B. Mulgrew, and P. Thomas, "Enabling self-configuration of fusion networks via scalable opportunistic sensor calibration," in *Proc. SPIE 10646, Signal Processing, Sensor/Information Fusion, and Target Recognition XXVII, 106460P*. Orlando, FL: SPIE, 2018.

[124] G. Marshall and D. Faulkner, "SAPIENT Interface Control Document," Dstl/Qinetiq, Tech. Rep., 2015.

[125] Y. Bar-Shalom and T. Fortmann, *Tracking and Data Association*. San Diego, CA: Academic Press Professional, 1987.

[126] 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, 2008.

[127] J. Hoffman and R. Mahler, "Multitarget miss distance via optimal assignment," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 34, no. 3, pp. 327 – 336, 2004.

[128] A. S. Rahmathullah, A. F. Garcia-Fernandez, and L. Svensson, "Generalized optimal sub-pattern assignment metric," in *20th International Conference on Information Fusion*. Xi'an, China: IEEE, 2017.

[129] X. R. Li, Z. Zhao, and X.-B. Li, "Evaluation of Estimation Algorithms: Credibility Tests," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 42, no. 1, 1 2012.

[130] E. P. Blasch, A. Rice, and C. Yang, "Nonlinear tracking evaluation using absolute and relative metrics," O. E. Drummond, Ed., 5 2006.

[131] Q. Liu, N. S. V. Rao, and X. Wang, "Staggered Scheduling of Sensor Estimation and Fusion for Tracking Over Long-Haul Links," *IEEE Sensors Journal*, vol. 16, no. 15, pp. 6130 – 6141, 2016.

[132] Q. Liu, X. Wang, N. S. V. Rao, K. Brigham, and B. V. K. V. Kumar, "Effect of Retransmission and Retrodiction on Estimation and Fusion in Long-Haul Sensor Networks," *IEEE/ACM Transactions on Networking*, vol. 24, no. 1, pp. 449 – 461, 2016.

[133] S. Zhang and Y. Bar-Shalom, "Optimal Update with Multiple Out-of-Sequence Measurements with Arbitrary Arriving Order," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 48, no. 4, pp. 3116 – 3132, 2012.

[134] S. R. Maskell, R. G. Everitt, R. Wright, and M. Briers, "Multi-target out-of-sequence data association: Tracking using graphical models," *Information Fusion*, vol. 7, no. 4, 12 2006.

[135] R. Mahler, "Approximate multisensor CPHD and PHD filters," in *13th International Conference on Information Fusion*. Edinburgh, UK: IEEE, 2010.

[136] S. Nannuru, S. Blouin, M. Coates, and M. Rabbat, "Multisensor CPHD filter," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 52, no. 4, pp. 1834 – 1854, 2016.

[137] K. V. Mardia and P. E. Jupp, *Directional Statistics*, 1st ed. John Wiley and Sons Inc., 1999.

[138] Kelvin Hughes, "Navigation Radar from Kelvin Hughes," 2017. [Online]. Available: https://www.kelvinhughes.com/maritime/commercial-ships/navigation-radar

[139] I. Schlangen, E. Delande, J. Houssineau, and D. Clark, "A PHD Filter with Negative Binomial Clutter," in *19th International Conference on Information Fusion (FUSION)*. Heidelberg, Germany: IEEE, 2016, pp. 658–665.

[140] D. Cormack and J. R. Hopgood, "Message Passing for Joint Registration and Tracking in Multistatic Radar," in *Sensor Signal Processing for Defence (SSPD) 2019*. Brighton, UK: IEEE, 2019.

[141] T. Richardson and R. Urbanke, *Modern Coding Theory*, 1st ed. New York, NY: Cambridge University Press, 2008.

[142] H. Wymeersch, J. Lien, and M. Z. Win, "Cooperative Localization in Wireless Networks," *Proceedings of the IEEE*, vol. 97, no. 2, pp. 427 – 450, 2009.

[143] L. Chen, M. Cetin, and A. Willsky, "Distributed Data Association For Multi-Target Tracking in Sensor Networks," in *8th International Conference on Information Fusion (FUSION)*. Philadelphia, PA: IEEE, 2005, pp. 9 – 16.

[144] D. Cormack and J. R. Hopgood, "Message Passing and Hierarchical Models for Simultaneous Tracking and Registration," *IEEE Transactions on Aerospace and Electronic Systems*, 2021.

[145] B. Huang and T. Jebara, "Approximating the Permanent with Belief Propagation," *ArXiv e-prints, ArXiv:0908.1769*, 2009.

[146] M. Chertkov and A. B. Yedidia, "Approximating the Permanent with Fractional Belief Propagation," *Journal of Machine Learning Research*, vol. 14, pp. 2029 – 2066, 2013.

[147] I. Schlangen, J. Franco, J. Houssineau, W. T. Pitkeathly, D. E. Clark, I. Smal, and C. Rickman, "Marker-Less Stage Drift Correction in Super-Resolution Microscopy Using the Single-Cluster PHD Filter," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 1, pp. 193 – 202, 2016.

[148] O. Hagen, J. Houssineau, I. Schlangen, E. Delande, J. Franco, and D. E. Clark, "Joint Estimation of Telescope Drift and Space Object Tracking," in *2016 IEEE Aerospace Conference*. Big Sky, MT: IEEE, 2016, pp. 1–10.

[149] E. Molina and Z. Zhu, "Persistent Aerial Video Registration and Fast Multi-View Mosaicing," *IEEE Transactions on Image Processing*, vol. 23, no. 5, pp. 2184 – 2192, 2014.

[150] D. Fox, "Adapting the Sample Size in Particle Filters Through KLD-Sampling," *The International Journal of Robotics Research*, vol. 22, no. 12, 12 2003.

[151] P. Thomas, J. Barr, B. Balaji, and K. White, "An open source framework for tracking and state estimation ('Stone Soup')," in *Proc. SPIE. 10200, Signal Processing, Sensor/Information Fusion, and Target Recognition XXVI*. Anaheim, CA: SPIE, 2017.

[152] P. Thomas, J. Barr, S. Hiscocks, C. England, S. Maskell, B. Balaji, and J. Williams, "Stone Soup: An Open-Source Framework for Tracking and State Estimation," *International Society for Information Fusion (ISIF) Perspectives*, pp. 14 – 19, 2 2019.

[153] D. Last, P. Thomas, S. Hiscocks, J. Barr, D. Kirkland, M. Rashid, S. B. Li, and L. Vladimirov, "Stone Soup: announcement of beta release of an open-source framework for tracking and state estimation," in *Proc. SPIE. 11018, Signal Processing, Sensor/Information Fusion, and Target Recognition XXVIII*. Baltimore, MD: SPIE, 2019.

[154] Z. Zhu and C. Taylor, "Conservative Uncertainty Estimation in Map-Based Vision-Aided Navigation," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 53, no. 2, pp. 941 – 949, 2017.

[155] S. J. Julier and J. K. Uhlmann, "Using covariance intersection for SLAM," *Robotics and Autonomous Systems*, vol. 55, no. 1, pp. 3 – 20, 2007.

[156] R. Streit, "The Probability Generating Functional for Finite Point Processes, and Its Application to the Comparison of PHD and Intensity Filters," *Journal of Advances in Information Fusion*, vol. 8, no. 2, pp. 119 – 132, 2013.

[157] D. Daley and D. Vere-Jones, *An introduction to the theory of point processes. vol. I. , Elementary theory and methods*, 2nd ed. New York: Springer, 2003.

[158] D. Clark and J. Houssineau, "Faa di Bruno's formula for chain differentials," *ArXiv e-prints, ArXiv:1310.2833*, pp. 1 – 7, 2013. [Online]. Available: https://arxiv.org/abs/1310.2833

[159] P. Bernhard, "Chain differentials with an application to the mathematical fear operator," *Nonlinear Analysis: Theory, Methods & Applications*, vol. 62, no. 7, pp. 1225 – 1233, 2005.

[160] S. N. Chiu, D. Stoyan, W. S. Kendall, and J. Mecke, *Stochastic geometry and its applications*. John Wiley & Sons, 2013.