Università degli Studi di Firenze
Facoltà di Ingegneria

Corso di laurea in
INGEGNERIA ELETTRICA E DELL'AUTOMAZIONE

Tracciamento multi-oggetto distribuito su reti di sensori

Distributed multitarget tracking over sensor networks

Tesi di Laurea di

Nicola Forti

Relatori:

Prof. Luigi Chisci

Prof. Giorgio Battistelli

Co-Relatore:

Ing. Claudio Fantacci

Aprile 2013

Anno Accademico 2011/2012

Contents

List of Figures

List of Tables

Acknowledgements

It gives me great pleasure in acknowledging the support and help of those who provided me with guidance and who, in one way or another contributed and extended their valuable assistance in the preparation and completion of this study.

First and foremost, my utmost gratitude goes to Prof. Luigi Chisci for providing me with direct supervision, encouragement and assistance throughout this study.

I must extend my gratitude to Claudio Fantacci, who spent many hours providing me with side-by-side assistance in the laboratory.

Last but not least, I would also like to thank my family and those who offered their support throughout the completion of my studies.

Chapter 1

Introduction

1.1 Multitarget Tracking Overview

Multitarget tracking (MTT) can be defined as the problem of tracking a number of targets through an area monitored by a single or multiple sensor nodes with embedded computing capabilities. Every sensor acquires noisy measurements of each target, and then computes estimates of the state of each target. Each target estimate is referred to as a *track*. Multitarget tracking is a challenging estimation problem of great practical relevance in an increasing number of civilian and military surveillance applications, including:

- air traffic control;
- coastal and maritime surveillance;
- traffic monitoring;
- border surveillance;
- localization of unmanned and automatic guided vehicles.

The widespread use and increasing sophistication of surveillance systems has stimulated great interest in algorithms capable of solving the multitarget

multisensor tracking problem. This is due to the fact that several critical issues are often encountered in practice when dealing with MTT, such as:

- missed detections, i.e. some targets may not be detected by a given sensor;
- false alarms (clutter), i.e. some measurements might not be provided by targets of interest;
- noisy measurements, i.e. all available measurements are subject to measurement noise;
- unknown origin of measurements, i.e. it is not known which target (if any) produced a given measurement;
- unknown and time-varying number of targets, i.e. new targets continuously appear in the scene and/or old targets disappear from the scene;
- possibly high density of targets and/or clutter;
- multiple sensors, i.e. a target can produce multiple measurements provided by different sensors;
- lack of observability from a single sensor, i.e. a single sensor may be unable to observe the full target state.

A combination of the aforementioned conditions can easily make the multitarget tracking task so critical that advanced techniques are required to solve the MTT problem.

A classical multitarget tracking system recursion can be schematized as in Figure 1.1. The tracker consists of four blocks, namely gating, data association, track maintenance and filtering-prediction. Track formation is responsible for

detecting new targets and initializing their tracks (initialization), as well as for recognizing disappeared targets and terminating their tracks (termination). Furthermore, it manages the confirmation logic that promotes tracks which will be displayed on screen. Unassociated measurements are used for track initialization, while associated observations are passed to the filtering and prediction block which, in turn, consists of a bank of filters, one for each track. They provide track predictions, used for gating and data association, as well as target estimates for track display. Moreover, the data association block, which aims at finding out the unknown source, either a target or clutter, of the available measurements, assigns measurements to the existing tracks. Finally, a gating block is cascaded at the input of data association in order to exclude measurements which are unlikely and thus reduce the overall computational burden.

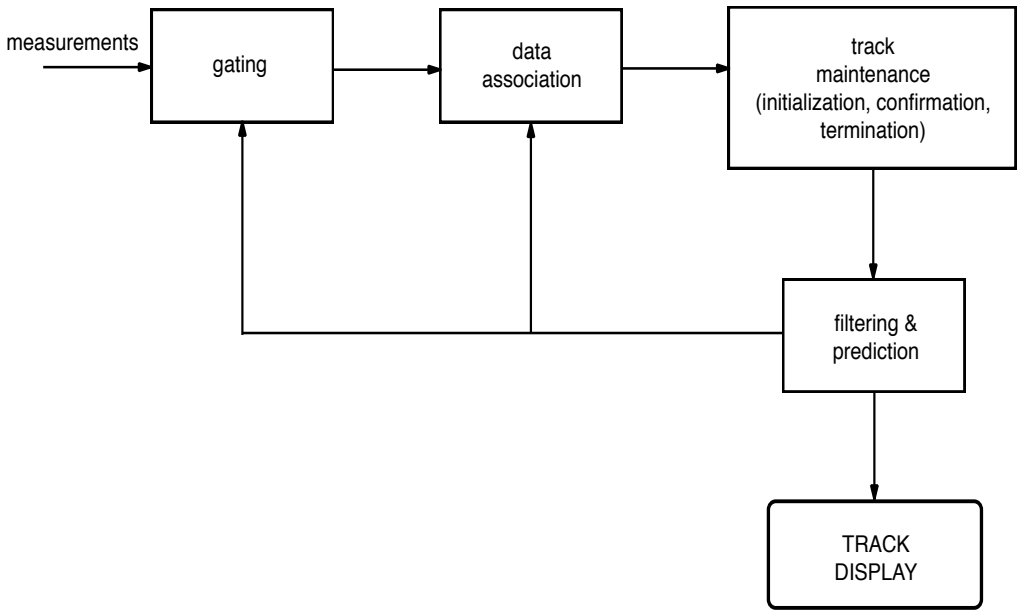

Figure 1.1: Basic elements of a conventional MTT system.

Recent advances in wireless sensor technology have opened up the possibility to develop efficient surveillance systems formed by the radio interconnection

of different low cost and low energy consumption devices with sensing, communication and processing capabilities. The availability of low-cost sensors have made it possible to create large-scale sensing that enables acquisition of massive data from spatially-distributed sources of information. This emerging technology introduces the following additional challenges in multitarget tracking:

- the single node has limited sensing and computational capabilities; in addition, also data transmission is limited, since it is the main responsible for energy consumption
- processing must be carried out in a distributed fashion, i.e. with no coordination of a central unit, and in a scalable way with respect to the network size;
- each node is unaware of the network topology.

The above considerations requires some sort of information fusion between multiple sensors, in order to counteract the multisensor tracking problem of incomplete measurements, that do not guarantee full state observability from a single node. Moreover, in order to solve large-scale information processing problems for sensor networks, development of novel algorithms that are scalable is required. In particular, decentralized approaches have been derived in order to solve estimation problems over sensor networks. They are preferable with respect to centralized architectures for the following reasons: first, the central processing node must run all the association and the estimation routines. Second, a massive amount of data is transmitted to the central node. Third, measurements may be arriving in an arbitrary order which further complicates the association performed by the central node.

In this thesis we address distributed multitarget tracking problem for sensor networks with a connected topology. The proposed distributed multitarget

tracking framework heavily relies on the *consensus* approach, which carries out a global fusion over the whole network, by iterating local fusion steps among neighboring sensors. This method ensures scalability.

1.2 Thesis Organization

The outline of the present thesis is as follows:

- Chapters 2 and 3 present basic concepts of recursive Bayesian estimation, with a particular emphasis on the nonlinear filter UKF.
- Chapters 4 and 5 describe the most commonly used target motion models and sensors, employed in tracking problems.
- Chapters 6 provides an overview on single target and multitarget data association problem, the focus is on single-scan Bayesian approaches. In addition, track initialization techniques and measurement validation are discussed.
- Chapter 7 introduces the multitarget multisensor problem. In particular, centralized architectures for multisensor data fusion are described.
- Chapter 8 proposes two solutions to distributed multitarget tracking problem over a sensor network.
- In Chapter 9 the performance of the proposed distributed algorithms will be assessed by means of simulation experiments on multisensor multitarget tracking scenarios.

Chapter 2

Recursive Bayesian Estimation

2.1 Kalman Filter

This chapter provides a practical introduction to the well-known Kalman filter. The Kalman filter addresses the problem of estimating the state x of a discrete-time dynamical system. In 1960, R.E. Kalman published his famous paper [21] presenting an efficient recursive solution to the discrete filtering and prediction problem, in particular their optimal solution in terms of Minimum Mean Square Error (MMSE). Consider a discrete-time linear system:

$$\begin{cases} x(k+1) = A(k)x(k) + B(k)u(k) + Dw(k) \\ z(k) = C(k)x(k) + v(k) \end{cases} \tag{2.1}$$

where: $x(k)$ is the state; $z(k)$ is the measurement; $u(k)$ is a known deterministic input; $w(k)$ and $v(k)$ represent the process and measurement noise, respectively. Both noises are assumed zero-mean, Gaussian and white i.e.

$$\begin{aligned} w(k) &\sim wn(0, Q(k)) \\ v(k) &\sim wn(0, R(k)) \end{aligned} \tag{2.2}$$

where

$$Q(k) = E\left[w(k)w(k)'\right] \tag{2.3}$$

is the process noise covariance and

$$R(k) = E\left[v(k)v(k)'\right] \tag{2.4}$$

is the measurement noise covariance. The matrices A, B, C, D, Q and R are assumed known and possibly time-varying. The initial state $x(0)$ is modeled as a random variable, Gaussian distributed with known mean and covariance. In addition, the two noise sequences and the initial state are assumed mutually independent. The aforementioned assumptions constitute the linear Gaussian assumption. Note that the linearity of (2.1) preserves the Gaussian property of the state and measurements (Gauss-Markov process).

The notation adopted throughout this thesis is the following: let Z^k denote the set of observations available at time k, i.e.

$$Z^k \triangleq \{z(s), s \leq k\} \tag{2.5}$$

then, the conditional mean

$$\hat{x}(t|k) \triangleq E\left[x(t)|Z^k\right] \tag{2.6}$$

depending on the values of t and k, is referred to as

- filtered estimate, if $t = k$
- smoothed estimate, if $t < k$
- predicted estimate, if $t > k$

The conditional covariance matrix of $x(t)$, given the data Z^k, is

$$P(t|k) \triangleq E\left[\left[x(t) - \hat{x}(t|k)\right]\left[x(t) - \hat{x}(t|k)\right]'|Z^k\right] \tag{2.7}$$

and represents the covariance of the estimation error.

The initial estimate of x(0), $\hat{x}(0|0)$, and the associated initial covariance $P(0|0)$, assumed to be available, are used to initialize the estimation algorithm. The

conditioning argument stands for Z^0, the initial (a priori) information. The algorithm consists of propagating the estimate $\hat{x}(k-1|k-1)$ and its covariance matrix $P(k-1|k-1)$, into the corresponding variables at the next time step, specifically $\hat{x}(k|k)$ and $P(k|k)$. This is due to the fact that the first two moments fully characterize a Gaussian random variable. The state estimation cycle is performed in two stages:

- **prediction** (or time update)
- **correction** (or measurement update)

The prediction equations are responsible for projecting forward in time the current state and error covariance estimates to obtain the *a priori* estimates for the next time step. The correction equations include a new measurement into the *a priori* estimate to obtain an improved *a posteriori* estimate and relative covariance. In each recursion, the process is repeated with the previous *a posteriori* estimate used to predict the new *a priori* estimate. This recursive nature is the key feature of the Kalman filter and gives the algorithm an essentially cyclic structure in which the same computations are performed at each time-step.

The estimation algorithm can be obtained in consequence of the above discussion. The predicted state $\hat{x}(k+1|k)$ follows by applying the operator of expectation conditioned on Z^k

$$E\left[x(k+1)|Z^k\right] = E\left[A(k)x(k) + B(k)u(k) + w(k)|Z^k\right] \tag{2.8}$$

since the process noise is white and zero-mean, it yields

$$\hat{x}(k+1|k) = A(k)\hat{x}(k|k) + B(k)u(k) \tag{2.9}$$

subtracting the above from (2.1), the state prediction error is obtained

$$\tilde{x}(k+1|k) \triangleq x(k+1) - \hat{x}(k+1|k) = A(k)\tilde{x}(k|k) + D(k)w(k) \tag{2.10}$$

The state prediction covariance is

$$\begin{aligned} &E\left[\tilde{x}(k+1|k)\tilde{x}(k+1|k)'|Z^k\right] = \\ &= A(k)E\left[\tilde{x}(k|k)\tilde{x}(k|k)'|Z^k\right]A(k)' + D(k)E\left[w(k)w(k)'\right]D(k)' \end{aligned} \tag{2.11}$$

that can be rewritten as

$$P(k+1|k) = A(k)P(k|k)A(k)' + D(k)Q(k)D(k)' \tag{2.12}$$

where the cross-terms in (2.11) are zero because the process noise is zero-mean and white, hence orthogonal to $\tilde{x}(k|k)$. The predicted measurement is obtained as follows

$$E\left[z(k+1)|Z^k\right] = E\left[C(k+1)x(k+1) + v(k+1)|Z^k\right] \tag{2.13}$$

since the measurement noise is zero-mean and white, one has

$$\hat{z}(k+1|k) = C(k+1)\hat{x}(k+1|k). \tag{2.14}$$

Then, subtracting (2.14) from the measurement equation (2.1), the measurement prediction error turns out to be

$$\tilde{z}(k+1|k) \triangleq z(k+1) - \hat{z}(k+1|k) = C(k+1)\tilde{x}(k+1|k) + v(k+1) \tag{2.15}$$

from which the resulting measurement prediction covariance is

$$S(k+1) = C(k+1)P(k+1|k)C(k+1)' + R(k+1) \tag{2.16}$$

Using (2.15), the covariance between state and measurement is

$$\begin{aligned} &E\left[\tilde{x}(k+1|k)\tilde{z}(k+1|k)'|Z^k\right] \\ &= E\left[\tilde{x}(k+1|k)\left[C(k+1)\tilde{x}(k+1|k) + v(k+1)\right]'|Z^k\right] \\ &= P(k+1|k)C(k+1)' \end{aligned} \tag{2.17}$$

Combining the above expression and (2.16), the filter gain at time k is

$$K(k) \triangleq P(k|k-1)C(k)'S(k)^{-1} \tag{2.18}$$

Therefore, the updated state estimate can be written as

$$\hat{x}(k|k) = \hat{x}(k|k-1) + K(k)\nu(k) \tag{2.19}$$

where $\nu(k)$ is the so called innovation defined as follows

$$\nu(k) \triangleq z(k) - \hat{z}(k|k-1) = \tilde{z}(k|k-1) \tag{2.20}$$

As a consequence, $S(k)$ is called the innovation covariance. Finally, the updated state covariance at time k is

$$\begin{aligned} P(k|k) &= P(k|k-1) - P(k|k-1)C(k)'S(k)^{-1}C(k)P(k|k-1) \\ &= [I - K(k)C(k)]\, P(k|k-1) \end{aligned} \tag{2.21}$$

or

$$P(k|k) = P(k|k-1) - K(k)S(k)K(k)' \tag{2.22}$$

There is also a recursion for the inverse covariance

$$P(k|k)^{-1} = P(k|k-1)^{-1} + C(k)'R(k)^{-1}C(k) \tag{2.23}$$

If the filter uses (2.23) instead of (2.21), it is called *information filter*. The filter gain can be rewritten as

$$K(k) = P(k|k)C(k)'R(k)^{-1} \tag{2.24}$$

The pseudo code in Algorithm 1 summarizes the Kalman filter equations.

Algorithm 1 KF

function KF($\hat{x}(0|-1)$, $P(0|-1)$)
  for all time $k = 0, 1, 2, \ldots$ **do**
    Correction
    $S(k) \longleftarrow R(k) + C(k)P(k|k-1)C(k)'$
    $K(k) \longleftarrow P(k|k-1)C(k)'S(k)^{-1}$
    $\nu(k) \longleftarrow z(k) - C(k)\hat{x}(k|k-1)$
    $\hat{x}(k|k) \longleftarrow \hat{x}(k|k-1) + K(k)\nu(k)$
    $P(k|k) \longleftarrow P(k|k-1) - K(k)S(k)K(k)'$
    Prediction
    $\hat{x}(k+1|k) \longleftarrow A(k)\hat{x}(k|k) + B(k)u(k)$
    $P(k+1|k) \longleftarrow A(k)P(k|k)A'(k) + D(k)Q(k)D(k)'$
  end for
  return prediction and correction sequence of $[\hat{x}, P]$
end function

It is important to point out that the covariance computations are independent of the state and measurements (and control, assumed known). Therefore, these calculations can be performed offline. To conclude, it is important to note that under the Gaussian assumption for the initial state and all the noises entering into the system, the Kalman filter is the optimal Minimum Mean Square Error (MMSE) state estimator. If the aforementioned random variables are not Gaussian, and only the first two moments are available, then the KF is the best linear unbiased MMSE estimator.

Chapter 3

Nonlinear Estimation

Unfortunately, in the vast majority of practical applications, the optimal linear Kalman filter, discussed in Chapter 2, cannot be applied. As a result, several approximations and suboptimal solutions have been proposed over the years. In this chapter the focus is on the estimation of the state of discrete-time nonlinear dynamical systems, observed via nonlinear measurements. More precisely, two different approaches will be discussed. Section 3.1 describes the extended Kalman filter (EKF), which exploits analytic approximations of nonlinear functions. In Section 3.2, another widely used nonlinear filter, called unscented Kalman filter, is presented. The latter belongs to the class of sampling methods, which approximate the posterior density by means of a set of samples.

3.1 Extended Kalman Filter

The Extended Kalman Filter [2, 16, 17] extends, through a linearization procedure, the use of the Kalman filter when the model of the dynamical

system is nonlinear. Consider the nonlinear state-space model:

$$\begin{cases} x(k+1) &= f(k, x(k)) + w(k) \\ z(k) &= h(k, x(k)) + v(k) \end{cases}$$

where $w(k)$ and $v(k)$ are independent zero-mean white Gaussian noise processes with covariance matrices Q and R, respectively. Notice that the noises are assumed additive. The basic idea of EKF is to approximate the nonlinear functions $f(\cdot)$ and $h(\cdot)$ via a first-order Taylor series expansion, at each time instant, around the most recent state estimate. Once a linear model is obtained, under the assumption that the conditional probability density function (pdf) is Gaussian, the standard Kalman filter can be applied to calculate the mean and the covariance of such a pdf. The resulting estimate turns out to be good only if the approximations (linearization and Gaussian distribution) are reasonable.

The approximation operates in two stages:

1. First, the following Jacobians are defined

$$\begin{cases} A(k) \cong \left[\frac{\partial f(k,\cdot)}{\partial x}\right]_{x=\hat{x}(k|k)} \\ C(k) \cong \left[\frac{\partial h(k,\cdot)}{\partial x}\right]_{x=\hat{x}(k|k-1)} \end{cases}$$

2. Once the above matrices $A(k)$ and $C(k)$ are evaluated, then a first-order Taylor approximation of the nonlinear functions $f(k, x(k))$ and $h(k, x(k))$ is performed around $\hat{x}(k|k)$ and $\hat{x}(k|k-1)$, respectively:

$$\begin{cases} f(k, x(k)) \cong f(k, \hat{x}(k|k)) + A(k)[x(k) - \hat{x}(k|k)] \\ h(k, x(k)) \cong h(k, \hat{x}(k|k-1)) + C(k)[x(k) - \hat{x}(k|k-1)] \end{cases}$$

Using these approximations the model obtained is now linear, consequently the Kalman filter can be applied to derive the following extended Kalman filter recursions:

- ***Update***

$$\begin{cases} S(k) = C(k)P(k|k-1)C(k)^{'} + R(k) \\ K(k) = P(k|k-1)C(k)^{'}S(k)^{-1} \\ \nu(k) = z(k) - h(k, \hat{x}(k|k-1)) \\ \hat{x}(k|k) = \hat{x}(k|k-1) + K(k)\nu(k) \\ P(k|k) = P(k|k-1) - K(k)S(k)K(k)^{'} \end{cases}$$

- ***Prediction***

$$\begin{cases} \hat{x}(k+1|k) = f(k, \hat{x}(k|k)) \\ P(k+1|k) = A(k)P(k|k)A(k)^{'} + Q(k) \end{cases}$$

Algorithm 2 EKF

function EKF($\hat{x}(0|-1)$, $P(0|-1)$, f, $\frac{\partial f}{\partial x}$, h, $\frac{\partial h}{\partial x}$)
  for all time $k = 0, 1, 2, \ldots$ **do**
    Correction
    $C(k) \longleftarrow \frac{\partial h}{\partial x}(\hat{x}(k|k-1))$
    $S(k) \longleftarrow R(k) + C(k)P(k|k-1)C(k)^{'}$
    $K(k) \longleftarrow P(k|k-1)C(k)^{'}S(k)^{-1}$
    $\nu(k) \longleftarrow z(k) - h(k, \hat{x}(k|k-1))$
    $\hat{x}(k|k) \longleftarrow \hat{x}(k|k-1) + K(k)\nu(k)$
    $P(k|k) \longleftarrow P(k|k) - K(k)S(k)K(k)^{'}$
    Prediction
    $A(k) \longleftarrow \frac{\partial f}{\partial x}(\hat{x}(k|k))$
    $\hat{x}(k+1|k) \longleftarrow f(k, \hat{x}(k|k))$
    $P(k+1|k) \longleftarrow A(k)P(k|k)A^{'}(k) + D(k)Q(k)D(k)^{'}$
  end for
  return prediction and correction sequence of $[\hat{x}, P]$
end function

The EKF recursion is summerized in Algorithm 2. The basic difference between EKF and KF is the evaluation of the Jacobians of the state transition and measurement equations. For the above reason, in the EKF the covariance calculations are no longer decoupled from the state estimate computations; hence, they cannot be carried out offline as in the Kalman filter. Note that since the Jacobians are evaluated at the estimated state rather than the true state, and higher order terms are neglected, the use of the series expansion introduces unmodeled errors. There is no certainty that even the second-order terms can compensate for such errors. These biases cannot be quantified and the resulting covariance matrix is not always accurate. As a consequence, the quality of the estimates is not guaranteed and the EKF is very sensitive to the accuracy of the initial conditions and of the available state space model.

3.2 Unscented Kalman Filter

In this section an alternative filter used for recursive nonlinear estimation is presented. This algorithm, referred to as the Unscented Kalman Filter (UKF), first proposed by Julier et al.[18], provides, in general, superior performance compared to EKF. The basic difference between the two approaches is the representation of Gaussian random variables for propagating through system dynamics. In the EKF, the state is approximated by a Gaussian distribution and the recursion is carried out by linearization of the nonlinear system. This can lead to large errors in the posterior mean and covariance of the transformed Gaussian random variable and consequently to suboptimal performance. In the UKF this problem is addressed by using a deterministic sampling approach. As in the EKF, the state distribution is approximated by a Gaussian random variable, but is now represented by a minimal set of sample points. These points must be selected so as to capture the mean and covariance of the Gaussian density and, when propagated through a nonlinear transformation,

they capture the posterior mean and covariance up to the second order of nonlinearity.

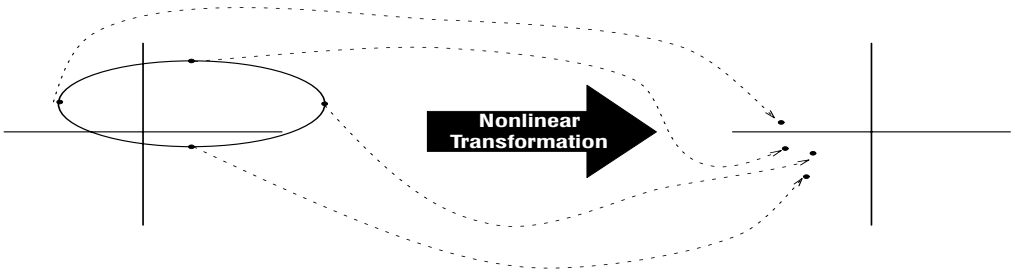

Figure 3.1: The principle behind the unscented transformation.

On the contrary, the EKF only provides first-order accuracy. Although no explicit Jacobian or Hessian calculations are necessary for the UKF, the computational complexity of the UKF is the same order as that of the EKF. The unscented Kalman filter technique approximates the distribution instead of the nonlinear system, by using the unscented transformation.

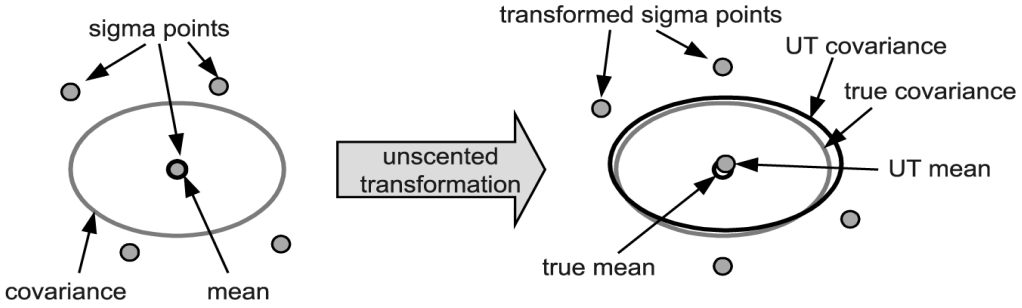

Figure 3.2: The approximation of mean and covariance via the unscented transformation.

3.2.1 Unscented Transformation

The Unscented Transformation (UT) [20, 30] is a method for calculating the statistics of a random variable which undergoes a nonlinear transforma-

tion. The basic idea is to approximate a probability distribution instead of a nonlinear function or transformation, which is more complicated. The approach is illustrated in Figure 3.1. A set of points called sigma points are chosen so that they have a certain mean and covariance. The nonlinear function is applied to each point to yield transformed points. The statistics of the transformed points can then be calculated to form an estimate of the nonlinearly transformed mean and covariance.

Consider propagating a random variable a through a nonlinear function $g : \mathbb{R}^n \to \mathbb{R}^{n_b}$, to produce a random variable b:

$$b = g(a) \tag{3.1}$$

Denote the mean and covariance of a as $\bar{a}$ and P_a respectively. In order to compute the statistics of $b = g(a)$, first the $2n + 1$ sigma points (n is the dimension of a) are first obtained as follows

$$\begin{aligned} a^{(0)} &= \bar{a} && i = 0 \\ a^{(i)} &= \bar{a} + \left(\sqrt{(n+\lambda)P_a}\right)_i && i = 1, \ldots, n \\ a^{(i)} &= \bar{a} - \left(\sqrt{(n+\lambda)P_a}\right)_i && i = n+1, \ldots, 2n \end{aligned} \tag{3.2}$$

where $\lambda = \alpha^2(n + \kappa) - n$ is a scaling parameter. In particular, the constant α represents the spread of the sigma points around $\bar{a}$ and is usually set equal to a small positive value, e.g. $1 \leq \alpha \leq 10^{-4}$. The constant κ is another design parameter, that is usually set equal to $3 - n$. Note that $(\sqrt{(n+\lambda)P_a})_i$ is the ith column of a matrix square root, obtained by Cholesky factorization or singular value decomposition of $(n + \lambda)P_a$. The sigma points in (3.2) are then propagated through the nonlinear function

$$b^{(i)} = g(a^{(i)}) \qquad\qquad i = 0, \ldots, 2n.$$

The mean and covariance for b are finally approximated using a weighted

sample mean and covariance of the posterior sigma points, by means of

$$\begin{aligned} \bar{b} &= \sum_{i=0}^{2n} w_m^{(i)} b^{(i)} \\ S_b &= \sum_{i=0}^{2n} w_c^{(i)} (b^{(i)} - \bar{b})(b^{(i)} - \bar{b})'. \end{aligned} \tag{3.3}$$

and the cross-covariance between $a^{(i)}$ and $b^{(i)}$ is

$$C_b = \sum_{i=0}^{2n} w_c^{(i)} (a^{(i)} - \bar{a})(b^{(i)} - \bar{b})'$$

where the weights $w_m^{(i)}$ and $w_c^{(i)}$, for the mean and, respectively covariance, are defined as follows:

$$\begin{aligned} w_m^{(0)} &= \frac{\lambda}{(n+\lambda)} \\ w_m^{(i)} &= \frac{1}{2(n+\lambda)} && i = 1, \ldots, 2n \\ w_c^{(0)} &= \frac{\lambda}{(n+\lambda)} + (1 - \alpha^2 + \beta) \\ w_c^{(i)} &= \frac{1}{2(n+\lambda)} && i = 1, \ldots, 2n \end{aligned} \tag{3.4}$$

The parameter β contains prior knowledge on the distribution of x and for Gaussian distributions is usually set equal to 2. Note that the Unscented Transformation approach results in approximations that are accurate to the third order for Gaussian inputs for all nonlinearities. For non-Gaussian inputs, approximations are accurate to at least the second order, and the accuracy of third and higher-order moments is influenced by the choice of α and β.

Furthermore, the UT can be rewritten in a convenient matrix form as follows:

$$\begin{aligned}
\mathcal{A} &= [\bar{a} \ \dots \ \bar{a}] + \sqrt{c}\left[0 \ , \sqrt{P_a} \ , -\sqrt{P_a}\right] \\
\mathcal{B} &= g(\mathcal{A}) \\
\bar{b} &= \mathcal{B} w_m \\
S_b &= \mathcal{B} W \mathcal{B}' \\
C_b &= \mathcal{A} W \mathcal{B}'
\end{aligned}$$

where $\mathcal{A}$ is the matrix of sigma points (placed as columns) and the function $g(\cdot)$ is separately applied to each column of the matrix. Moreover $c = \alpha^2(n + \kappa)$, while the vector w_m and the matrix W are defined as folllows:

$$\begin{aligned}
w_m &= \left[w_m^{(0)} \dots w_m^{(2n)}\right]' \\
W &= (I - [w_m \dots w_m]) \times diag\left(w_c^{(0)} \dots w_c^{(2n)}\right) \times (I - [w_m \dots w_m])'
\end{aligned}$$

3.2.2 Unscented Kalman Filter Algorithm

The Unscented Kalman Filter (UKF) is a simple extension of the UT to recursive state estimation for the following stochastic dynamical system (assumed for simplicity with additive noise)

$$\begin{cases}
x(k+1) = f(k, x(k)) + w(k) \\
z(k) = h(k, x(k)) + v(k)
\end{cases}$$

where $x(k) \in \mathbb{R}^n$ is the state, $z(k) \in \mathbb{R}^m$ is the measurement, $w(k) \sim wn(0, Q(k))$ is the process noise and $v(k) \sim wn(0, R(k))$ is the measurement noise.

Exploiting the unscented transformation in the previously presented matrix form, the *Prediction* and the *Correction* steps are carried out as follows:

- ***Prediction***:

Starting from the sigma points, the mean $\hat{x}(k|k-1)$ and the covariance $P(k|k-1)$ are calculated:

$$\begin{aligned}
\mathcal{X}(k-1) &= [\hat{x}(k-1|k-1) \ \dots \ \hat{x}(k-1|k-1)] + \\
&\quad + \sqrt{c}\left[0 \ , \sqrt{P(k-1|k-1)} \ , -\sqrt{P(k-1|k-1)}\right] \\
\hat{\mathcal{X}}(k|k-1) &= f(k-1, \mathcal{X}(k-1)) \\
\hat{x}(k|k-1) &= \hat{\mathcal{X}}(k|k-1) w_m \\
P(k|k-1) &= \hat{\mathcal{X}}(k|k-1) W \hat{\mathcal{X}}(k|k-1)^{'} + Q(k-1) \\
\mathcal{X}(k|k-1) &= [\hat{x}(k|k-1) \ \dots \ \hat{x}(k|k-1)] + \sqrt{c}\left[0 \ , \sqrt{P(k|k-1)} \ , -\sqrt{P(k|k-1)}\right] \\
Z(k|k-1) &= h(k, \mathcal{X}(k|k-1)) \\
\hat{z}(k|k-1) &= Z(k|k-1) w_m
\end{aligned}$$

- ***Correction***:

  the innovation covariance $S(k)$ and the cross-covariance between state and measurement $T(k)$ are obtained:

$$\begin{aligned}
S(k) &= Z(k|k-1) W Z(k|k-1)^{'} + R(k) \\
T(k) &= \mathcal{X}(k|k-1) W Z(k|k-1)^{'}
\end{aligned}$$

  Finally, the filter gain $K(k)$, the updated state $\hat{x}(k|k)$ and its covariance matrix $P(k|k)$ are calculated by:

$$\begin{aligned}
K(k) &= T(k) S(k)^{-1} \\
\hat{x}(k|k) &= \hat{x}(k|k-1) + K(k)\left[z(k) - \hat{z}(k|k-1)\right] \\
P(k|k) &= P(k|k-1) - K(k) S(k) K(k)^{'}
\end{aligned}$$

Algorithm 3 UKF

function UKF($\hat{x}(0|0)$, $P(0|0)$, f, h, α, β, κ)

Weights computation

$n \longleftarrow$ ***size(***$\hat{x}_{0|0}$***)***

$\lambda \longleftarrow \alpha^2(n+\kappa) - n$

$w_m^{(0)} \longleftarrow \frac{\lambda}{(n+\lambda)}$

$w_m^{(1,\dots,2n)} \longleftarrow \frac{1}{2(n+\lambda)}$

$w_c^{(0)} \longleftarrow \frac{\lambda}{(n+\lambda)} + (1 - \alpha^2 + \beta)$

$w_c^{(1,\dots,2n)} \longleftarrow \frac{1}{2(n+\lambda)}$

$w_m \longleftarrow \left[w_m^{(0)} \dots w_m^{(2n)}\right]'$

$W \longleftarrow (I - [w_m \dots w_m]) \times diag\left(w_c^{(0)} \dots w_c^{(2n)}\right) \times (I - [w_m \dots w_m])'$

$c \longleftarrow \alpha^2(n+\kappa)$

for all time $t = 1, 2, \dots$ **do**

Sigma points computation

$[U, \Sigma] \longleftarrow$ ***svd(***$P(k-1|k-1)$***)***

$T(k-1|k-1) \longleftarrow U\sqrt{\Sigma}$

$$\mathcal{X}(k-1) \longleftarrow \left[\hat{x}(k-1|k-1) \ \dots \ \hat{x}(k-1|k-1)\right] + \\ + \sqrt{c}\left[0 \ , T(k-1|k-1), -T(k-1|k-1)\right]$$

$\hat{\mathcal{X}}(k|k-1) \longleftarrow f(k-1, \mathcal{X}(k-1))$

Prediction

$\hat{x}(k|k-1) \longleftarrow \hat{\mathcal{X}}(k|k-1) w_m$

$P(k|k-1) \longleftarrow \hat{\mathcal{X}}(k|k-1) W \hat{\mathcal{X}}(k|k-1)' + Q(k-1)$

$[U, \Sigma] \longleftarrow$ ***svd(***$P(k|k-1)$***)***

$T(k|k-1) \longleftarrow U\sqrt{\Sigma}$

20:

$$\mathcal{X}(k|k-1) \longleftarrow \Big[\hat{x}(k|k-1) \ \dots \ \hat{x}(k|k-1)\Big] + \\ + \sqrt{c}\Big[0 \ , T(k|k-1), -T(k|k-1)\Big]$$

21: $Z(k|k-1) \longleftarrow h(k, \mathcal{X}(k|k-1))$

Correction

22: $S(k) \longleftarrow Z(k|k-1)WZ(k|k-1)^{'} + R(k)$

23: $T(k) \longleftarrow \mathcal{X}(k|k-1)WZ(k|k-1)^{'}$

24: $K(k) \longleftarrow T(k)S(k)^{-1}$

25: $\hat{x}(k|k) \longleftarrow \hat{x}(k|k-1) + K(k)\left[z(k) - \hat{z}(k|k-1)\right]$

26: $P(k|k) \longleftarrow P(k|k-1) - K(k)S(k)K(k)^{'}$

27: **end for**

28: **return** prediction and correction sequence of $[\hat{x}, P]$

29: **end function**

Chapter 4

Target Kinematic Models

This chapter presents some of the most commonly used target motion models. White noise acceleration (WNA) models are defined by setting the second-order derivative of the position to zero and modeling the disturbances as random inputs, so that acceleration becomes a white process noise. Moreover, we consider the Wiener process acceleration model, in which the third-order derivative of the position is modeled as a white noise. Finally, Section 4.3 discusses the exponentially correlated acceleration model. Although the discussion deals with one-dimensional motion, for 2- or 3-dimensional motions, one can consider independent 1-dimensional models for each coordinate.

4.1 White Noise Acceleration

The white noise acceleration model is 2-nd order for each coordinate. Consider an object moving with constant velocity in a coordinate ξ. Then, in the absence of noise, the equation of motion is

$$\ddot{\xi}(t) = 0 \tag{4.1}$$

if process noise is included to take into account, the (4.1) becomes

$$\ddot{\xi}(t) = \tilde{w}(t) \tag{4.2}$$

where $\tilde{w}$ is a continuous time zero-mean white noise with

$$E\left[\tilde{w}(t)\right] = 0 \tag{4.3}$$

$$E\left[\tilde{w}(t)\tilde{w}(\tau)\right] = \tilde{q}(t)\delta(t-\tau) \tag{4.4}$$

where $\tilde{q}$ is the intensity of $\tilde{w}$. To express (4.2) in state-space form, the 2-dimensional state vector of the continuous white noise acceleration (CWNA) model becomes:

$$x = [\xi \quad \dot{\xi}]' \tag{4.5}$$

Note that the velocity is a Wiener process, i.e. the integral of white noise. The state equation is

$$\dot{x}(t) = Ax(t) + D\tilde{w}(t) \tag{4.6}$$

where

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \qquad D = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{4.7}$$

Since, in general, the observations are performed in discrete time, the corresponding discrete-time state equations are considered:

$$x(k+1) = A_d x(k) + w(k) \tag{4.8}$$

where

$$A_d = e^{AT} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \tag{4.9}$$

and the discrete-time process noise is

$$w(k) = \int_0^T e^{A(T-\tau)} D\tilde{w}(kT+\tau)\, d\tau \tag{4.10}$$

Thus, assuming $\tilde{q}$ to be constant, from (4.4) the covariance of $w(k)$ is obtained as follows

$$Q = E[w(k)w(k)'] = \int_0^T e^{Ar} DQD' e^{A'r} dr = \begin{bmatrix} \frac{1}{3}T^3 & \frac{1}{2}T^2 \frac{1}{2}T^2 & T \end{bmatrix} \tilde{q} \tag{4.11}$$

where $\tilde{q}$ is a design parameter that, if small enough, leads to a nearly constant velocity (NCV) model. For this model, the process noise intensity $\tilde{q}$ is chosen so that $\sqrt{Q_{22}} = \sqrt{\tilde{q}T}$ quantifies the fluctuations of velocity within a sampling interval.

In some cases, it is convenient to define a direct discrete-time model rather than a discretized version of a continuous-time model. In such cases the process noise, also modeled as white, enters through a noise gain, denoted as D. The discrete-time process noise $w(k)$ is a scalar zero-mean white sequence

$$E\left[w(k)w(j)\right] = \sigma_w^2 \delta_{kj} \tag{4.12}$$

where δ_{kj} is the Kronecker delta. The dynamic equation is the following

$$x(k+1) = Ax(k) + Dw(k) \tag{4.13}$$

where D is the noise gain. Moreover, in second-order models one has

$$\tilde{w}(t) = w(k) \qquad t \in [kT, (k+1)T] \tag{4.14}$$

with accelerations uncorrelated from one period to another. The signal defined in (4.14) represents a piecewise constant acceleration, which is the approximation used to model uncertainties in the DWNA model. The transition matrix in (4.13) is

$$A = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \tag{4.15}$$

Since the velocity increment during the k-th sampling period is $w(k)T$ and the acceleration effect on the position is $w(k)\frac{1}{2}T^2$, the vector multiplying $w(k)$ in (4.13) is

$$D = \begin{bmatrix} \frac{1}{2}T^2 \\ T \end{bmatrix} \tag{4.16}$$

Hence, the covariance of $Dw(k)$ is

$$Q = E\left[Dw(k)(Dw(k))'\right] = D\sigma_w^2 D' = \begin{bmatrix} \frac{1}{4}T^4 & \frac{1}{2}T^3 \\ \frac{1}{2}T^3 & T^2 \end{bmatrix} \sigma_w^2 \tag{4.17}$$

Usually σ_w is chosen in the range $0.5a_M \leq \sigma_w \leq a_M$, where a_M is the maximum acceleration magnitude. This model becomes a nearly constant velocity (NCV) model if the changes in velocity over a sampling period, which are the same order of $\sigma_w T$, are small with respect to the actual velocity.

4.2 Wiener Process Acceleration

The Wiener process acceleration model is 3-rd order for each motion coordinate. The equation of motion of a constant acceleration object with disturbances modeled by a continuous-time zero-mean white noise for coordinate ξ is

$$\dddot{\xi}(t) = \tilde{w}(t) \tag{4.18}$$

the continuous-time state equation is

$$\dot{x}(t) = Ax(t) + D\tilde{w}(t) \tag{4.19}$$

where

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \qquad D = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{4.20}$$

and

$$x = \begin{bmatrix} \xi & \dot{\xi} & \ddot{\xi} \end{bmatrix}' \tag{4.21}$$

Since the acceleration is a Wiener process, the model is called continuous Wiener process acceleration (CWPA) or white noise jerk model (jerk is the derivative of the acceleration).

The resulting discrete-time state equation is

$$x(k+1) = A_d x(k) + w(k) \tag{4.22}$$

with the transition matrix

$$A_d = \begin{bmatrix} 1 & T & \frac{1}{2}T^2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix} \tag{4.23}$$

and the covariance matrix of $w(k)$

$$Q = E\left[w(k)w(k)'\right] = \begin{bmatrix} \frac{1}{20}T^5 & \frac{1}{8}T^4 & \frac{1}{6}T^3 \\ \frac{1}{8}T^4 & \frac{1}{3}T^3 & \frac{1}{2}T^2 \\ \frac{1}{6}T^3 & \frac{1}{2}T^2 & T \end{bmatrix} \tilde{q} \tag{4.24}$$

For this model, the process noise intensity $\tilde{q}$ is chosen so that $\sqrt{Q_{33}} = \sqrt{\tilde{q}T}$, quantifies the fluctuations of acceleration within a sampling interval.

4.3 Exponentially Correlated Acceleration

The exponentially correlated acceleration model is one of the most widely used to track maneuvering targets. This model, first introduced by Singer [29], is based on the fact that vehicles under consideration generally follow constant velocity trajectories, but they can also deviate from this straight line trajectories. To this end, the model also accounts for this target maneuver capability.

Consider a target that normally moves at constant velocity, with perturbations on the constant velocity trajectory due to turns, evasive maneuvers and atmospheric turbulence. In a single coordinate, the equation of motion of such a target can be represented by

$$\dot{x}(t) = Ax(t) + Ga(t) \tag{4.25}$$

where the components of $x(t)$ are the target position at time t and its speed, $a(t)$ is the acceleration at time t, also referred to as the target maneuver variable. Moreover

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \qquad G = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{4.26}$$

The target acceleration is correlated in time. Specifically, a target accelerating at time t, is likely to be accelerating also at time $t + \tau$ for sufficiently small τ. For example, lazy turns cause correlated acceleration inputs up to one minute, atmospheric turbulence for one or two seconds, and evasive maneuvers can provide correlated acceleration inputs for intervals between ten and thirty seconds. The correlation function associated with the target acceleration is typically modeled as

$$r(\tau) = E\left[a(t)a(t+\tau)\right] = \sigma_m^2 e^{-\alpha|\tau|} \qquad , \alpha \geq 0 \tag{4.27}$$

where σ_m^2, computed using the model in figure 4.1, is the instantaneous variance of the acceleration and α is the reciprocal of the maneuver time constant. Typical values are:

- $\alpha \simeq \dfrac{1}{60}$ for a lazy turn
- $\alpha \simeq \dfrac{1}{20}$ for an evasive maneuver
- $\alpha \simeq 1$ for atmospheric turbulence

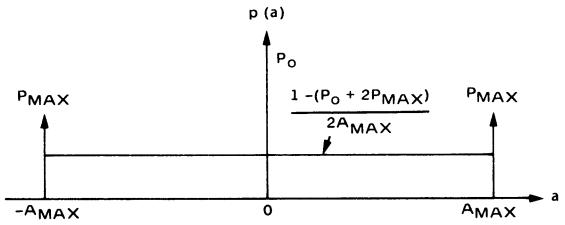

Figure 4.1: Model of the target acceleration probability density.

The correlation function is shown in figure 4.2.

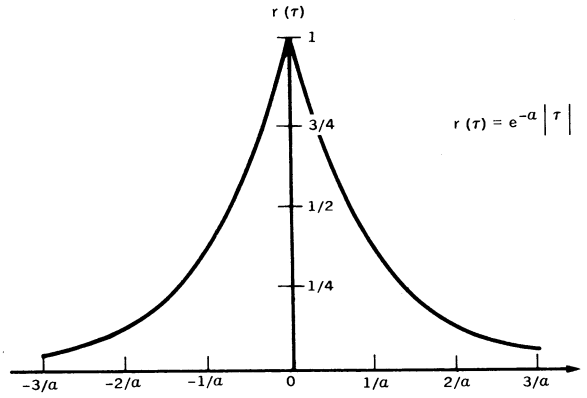

Figure 4.2: Correlation function of target acceleration.

Due to the acceleration being correlated in time, the maneuvers are neither additive nor Gaussian. Singer's probability density function for target maneuvers is shown in figure 4.1. As illustrated in the figure, the maximum target acceleration rate is $a_{max}(-a_{max})$ with probability p_{max}, whereas the target has no acceleration with probability p_0 and accelerates between $-a_{max}$ and a_{max} according to the uniform distribution. In order to use this model in an optimal filter such as a Kalman filter, the maneuver noise needs to be whitened. Singer uses a method analogous to the Wiener-Kolmogorov whitening procedure. The state vector is augmented to include the maneuver variables.

Using the correlation function $r(\tau)$ in (4.27), the acceleration $a(t)$ can be expressed in terms of white noise by the first-order Markov process

$$\dot{a}(t) = -\alpha a(t) + w(t) \tag{4.28}$$

driven by the white noise $w(t)$, with

$$E\left[w(t)w(\tau)\right] = 2\alpha\sigma_m^2\delta(t-\tau) \tag{4.29}$$

Note that as α increases, the process $a(t)$ becomes uncorrelated faster. For $\alpha \to \infty$ and $\sigma_m \to \infty$, the acceleration becomes white noise. This corresponds

to the white noise acceleration model (WNA), which is second order. For $\alpha \to 0$, $a(t)$ becomes the integral of white noise, i.e. the acceleration becomes a Wiener process. This case corresponds to the Wiener process acceleration (WPA) model, which is third order.

The target equation of motion can be rewritten in terms of the white noise $w(t)$ as

$$\dot{x}(t) = Ax(t) + Gw(t) \tag{4.30}$$

where the components of $x(t)$ are the target position, speed and acceleration at time t, and

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & -\alpha \end{bmatrix} \qquad G = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{4.31}$$

The discrete-time equation corresponding to (4.30) for sampling period T is

$$x(k+1) = A_d x(k) + w(k) \tag{4.32}$$

where

$$A_d = e^{AT} = \begin{bmatrix} 1 & T & (\alpha T - 1 + e^{-\alpha T})/\alpha^2 \\ 0 & 1 & (1 - e^{-\alpha T})/\alpha \\ 0 & 0 & e^{-\alpha T} \end{bmatrix} \tag{4.33}$$

Assuming T sufficiently small, so that $\alpha T \ll 1$, the covariance matrix of the discrete-time process noise $w(k)$ turns out to be:

$$Q = 2\alpha\sigma_m^2 \begin{bmatrix} T^5/20 & T^4/8 & T^3/6 \\ T^4/8 & T^3/3 & T^2/2 \\ T^3/6 & T^2/2 & T \end{bmatrix} \tag{4.34}$$

The above equation is adequate under the assumption that the sampling interval T is much less than the time constant $1/\alpha$ of the maneuver autocorrelation. The exact expression of Q has the following elements

$$q_{11} = \frac{\sigma_m^2}{\alpha^4}\left[1 - e^{-2\alpha T} + 2\alpha T + \frac{2\alpha^3 T^3}{3} - 2\alpha^2 T^2 - 4\alpha T e^{-\alpha T}\right] \tag{4.35}$$

$$q_{12} = \frac{\sigma_m^2}{\alpha^3}\left[e^{-2\alpha T} + 1 - 2e^{\alpha T} + 2\alpha T e^{-\alpha T} - 2\alpha T + \alpha^2 T^2\right] \tag{4.36}$$

$$q_{13} = \frac{\sigma_m^2}{\alpha^2}\left[1 - e^{-2\alpha T} - 2\alpha T e^{-\alpha T}\right] \tag{4.37}$$

$$q_{22} = \frac{\sigma_m^2}{\alpha^2}\left[4e^{-\alpha T} - 3 - e^{-2\alpha T} + 2\alpha T\right] \tag{4.38}$$

$$q_{23} = \frac{\sigma_m^2}{\alpha}\left[e^{-2\alpha T} + 1 - 2e^{-\alpha T}\right] \tag{4.39}$$

$$q_{33} = \sigma_m^2\left[1 - e^{-2\alpha T}\right] \tag{4.40}$$

Chapter 5

Tracking sensors

In this chapter, different sensors commonly employed in target tracking are briefly reviewed. The focus is on the basic concepts of sensor operational characteristics, and on the measurement equations. Sensor models are discussed from the perspective of a designer of tracking systems rather than a sensor designer. Kinematic measurements are of particular interest, because they give information about the existence and location of the target, usually relative to the sensor. Typical measurements include range, range rate, bearing (azimuth) and elevation angles. They are the output of the signal processing, provided by the sensor, usually referred to as measurements, observations, hits or plots. In tracking problems, several kinematic sensors are used, but each one can be generally described by the following measurement equation

$$\begin{aligned} z(k) &= h(k, x(k)) + v(k) \\ v(k) &\sim wn(0, R) \end{aligned} \tag{5.1}$$

where $x(k)$ is the kinematic state, $z(k)$ is the measurement provided by the sensor, $v(k)$ is the measurement noise and $h(\cdot)$ is the measurement function. In order to track the targets of interest and achieve good performance, the concept of observability of the kinematic state provided by a given sensor is

crucial. A discrete-time system

$$\begin{aligned} x(k+1) &= f(k, x(k)) + w(k) \\ z(k) &= h(k, x(k)) + v(k) \end{aligned} \tag{5.2}$$

is observable whenever in the absence of noise i.e. for $w(k) \equiv 0$ and $v(k) \equiv 0$, it is possible to uniquely determine $x(0)$ given $y(0), y(1), \ldots$. In the sequel, observability will be described for each presented sensor.

5.1 RADAR

RADAR is a sensor for the detection and location of reflecting objects with the following operating principle

- The radar transmits a pulse of Radio Frequency (RF) energy from an antenna that propagates in space.
- A part of the transmitted energy is intercepted by a reflecting object, the target, located at a certain distance from the radar.
- The energy intercepted by the target is reflected in many directions.
- Some of the reflected energy, called echo, returns to the radar, Δt seconds later.
- The returning pulse is amplified by a receiver that decides whether or not a target echo is present.
- If the target has been detected, its information is acquired.

Modern tracking radars are capable of measuring target angle (azimuth and elevation), range and range rate. The range to a target is calculated by measuring the time it takes for the radar pulse to propagate at the speed of light, intercept the target and return back to the radar. Among all types of

sensors, the radar provides the most accurate measurement of the distance to a target at long range. At short ranges, the precision can be a few centimeters. The range measurement accuracy depends on the radar signal bandwidth, and is determined as follows

$$R = \frac{c\,\Delta t}{2} \tag{5.3}$$

where c is the speed of light. In addition, the radar measures the azimuth angle. It can be determined as the angle where the magnitude of the reflected signal from a scanning antenna is at the maximum. Air-surveillance radars with rotating antenna beams, determine the direction to a target in this manner. This usually requires an antenna with a narrow beamwidth.

Range and angle measurements provide a 2D relative position of the target from the radar location. The corresponding measurement equation for sensor i, used in tracking algorithms, is

$$z^i(k) = \begin{bmatrix} r^i(k) \\ \theta^i(k) \end{bmatrix} + v^i(k) = h^i(x(k)) + v^i(k) \tag{5.4}$$

where the range is

$$r^i = \sqrt{(p_x - p_x^i)^2 + (p_y - p_y^i)^2} \tag{5.5}$$

and $p^i = (p_x^i, p_y^i)$ is the position of sensor i in Cartesian coordinates. The angle component of the measurement is obtained as

$$\theta^i = \angle[(p_x - p_x^i) + j(p_y - p_y^i)] \tag{5.6}$$

Usually, both range and azimuth errors are assumed independent, so that $R = diag\{\sigma_r^2, \sigma_\theta^2\}$ where σ_r and σ_θ are the standard deviations which depend on the characteristics of the employed sensor. It is of interest to note that a single RADAR can exactly locate the position of a target in absence of noise and, hence, the observability of the kinematic state of the target is guaranteed.

5.2 Time Of Arrival (TOA)

The Time Of Arrival (TOA) sensor measures the distance between the target and the sensor as the time it takes for the electromagnetic or acoustic signal to cover that distance. Denoted the known position of sensor i as $p^i = (p_x^i, p_y^i)$, then the measurement function for a TOA sensor is

$$h^i(x) = \sqrt{(p_x - p_x^i)^2 + (p_y - p_y^i)^2} \tag{5.7}$$

To achieve observability of the kinematic state, at least three TOA sensors are necessary in a two-dimensional space, four in a three-dimensional space.

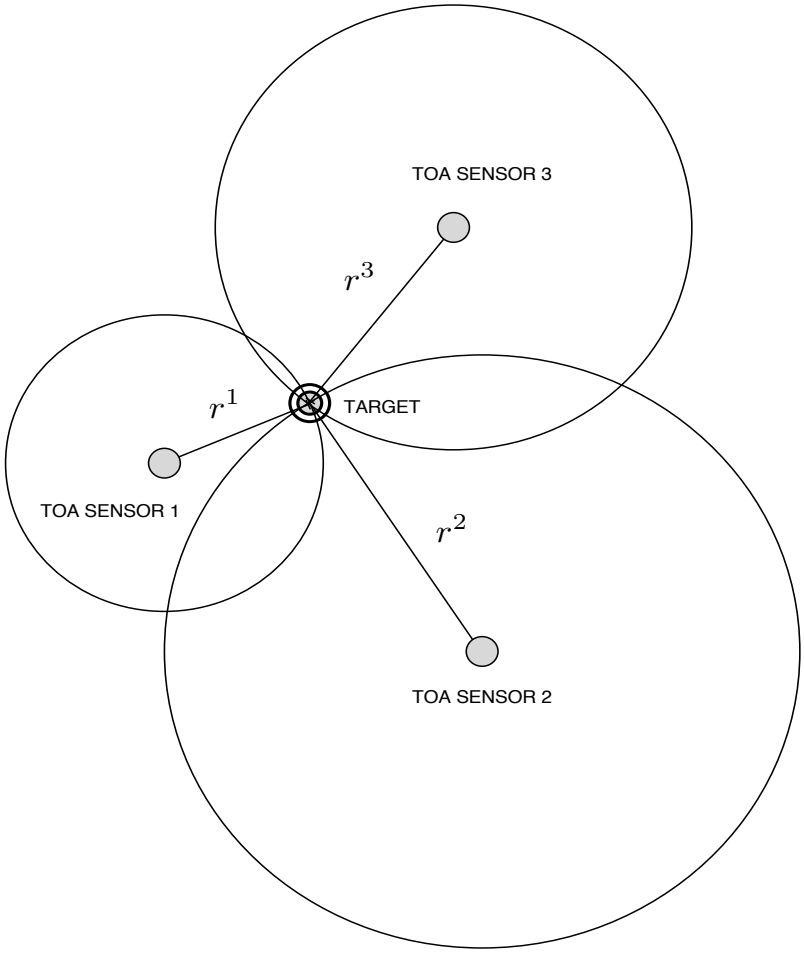

Figure 5.1: Observability with three TOA sensors.

5.3 Direction Of Arrival (DOA)

The Direction Of Arrival (DOA) sensor, also called Angle Of Arrival (AOA), measures the angular direction of the signal returning from the target of interest. The measurement function for a DOA sensor i is

$$h^i(x) = \angle[(p_x - p_x^i) + j(p_y - p_y^i)] \tag{5.8}$$

Observability of the kinematic state in a two-dimensional space is guaranteed with at least two DOA sensors.

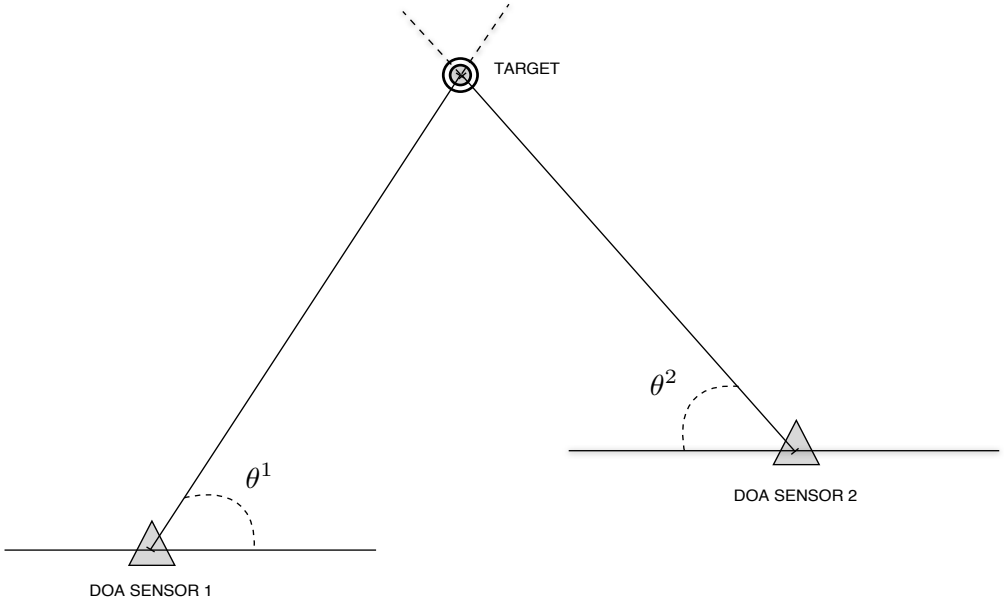

Figure 5.2: Observability with two DOA sensors.

5.4 Doppler

Doppler sensors rely on the Doppler effect to enhance target detection. The Doppler effect causes a frequency shift when there is a relative range rate, or radial velocity, between the target and the sensor. As the transmitter signal is reflected from a target, the carrier frequency of the returning signal is shifted.

Consider a monostatic radar (both transmitter and receiver are co-located) so that the round-trip distance is twice the distance between the transmitter and the target. The Doppler frequency shift, denoted as f_d, is a function of the carrier wavelength λ and the relative radial velocity (range rate) between the radar and the target , $\dot{r}$, and is given by

$$f_d = -\frac{2\dot{r}}{\lambda} \tag{5.9}$$

where $\lambda = c/f$ is the wavelength, c is the light speed and f is the carrier frequency. Note that a negative sign is included in (5.9) to account for the fact that, whenever the target is moving away from the radar, the relative radial velocity is defined to be positive and results in a negative frequency shift. For a Doppler sensor i, the measurement equation (5.1) can be written as

$$z^i(k) = \dot{r}^i(k) + v^i(k) \tag{5.10}$$

As a consequence of (5.9), the frequency shift, that can be easily be measured, is proportional to the range rate

$$\dot{r}^i = \frac{d}{dt}\left[\sqrt{(p_x - p_x^i)^2 + (p_y - p_y^i)^2 + (p_z - p_z^i)^2}\right] \tag{5.11}$$

Assuming the position of the sensor i (p_x^i, p_y^i, p_z^i) and its velocity, if the sensor is moving, $(\dot{p}_x^i, \dot{p}_y^i, \dot{p}_z^i)$, the expression becomes the following

$$\dot{r}^i = \frac{(p_x - p_x^i)(\dot{p}_x - \dot{p}_x^i) + (p_y - p_y^i)(\dot{p}_y - \dot{p}_y^i) + (p_z - p_z^i)(\dot{p}_z - \dot{p}_z^i)}{\sqrt{(p_x - p_x^i)^2 + (p_y - p_y^i)^2 + (p_z - p_z^i)^2}} \tag{5.12}$$

Note that, among all the different sensors mentioned in this section, only the measurement provided by the Doppler sensor depends on the velocity components $(\dot{p}_x, \dot{p}_y, \dot{p}_z)$ as well as the position components (p_x, p_y, p_z) of the target of interest. Observability analysis on Doppler sensors has shown that the kinematic state of a target in a two-dimensional space can be estimated using at least three not aligned sensors. In three dimensions, at least four sensors, not coplanar, are necessary.

5.5 Clutter Model

Clutter is a term that refers to unwanted echoes returning to the radar (or another active sensor) from different sources e.g. ground, sea, weather, buildings, etc. These spurious reflectors and the sensor noise cause false alarms. As a matter of fact, a sensor with N_{RC} resolution cells, declares a detection in a cell if the reflected amplitude of the signal exceeds a given threshold. For this reason, false alarm detections can occur when the sensor points to a region where there is no target, but receives a signal due to sensor noise or clutter.

Under the following assumptions:

- the detection events in each cell are mutually independent;
- the probability of false alarm detection is P_{FA} in each cell;

the probability mass function (pmf) of the number of false alarms, in these N_{RC} cells, can be obtained as the probability of the number of successes in a sequence of N_{RC} independent Bernoulli experiments, each of which yields success with probability P_{FA} (binomial distribution). Then

$$Prob\{n_{FA} = m\} = \mu_{FA}(m) = \binom{N_{RC}}{m} (P_{FA})^m (1 - P_{FA})^{N_{RC}-m} \tag{5.13}$$

Consequently, the spatial density of the false alarms is

$$\lambda = \frac{E\left[n_{FA}\right]}{V} = \frac{N_{RC} P_{FA}}{V} \tag{5.14}$$

where V is the volume of the cells. Under the condition

$$P_{FA} \ll 1 \tag{5.15}$$

(5.13) can be approximated by the Poisson distribution, for N_{RC} large enough so that $N_{RC}P_{FA}$ is the same order of magnitude of 1. Then, the pmf $\mu_{FA}(m)$ becomes

$$\mu_{FA}(m) = e^{-N_{RC}P_{FA}} \frac{(N_{RC}P_{FA})^m}{m!} \tag{5.16}$$

Combining (5.14) and (5.16), the probability mass function of the number of false alarms in the volume V, with respect to their spatial density, is

$$\mu_{FA}(m) = e^{-\lambda V} \frac{(\lambda V)^m}{m!} \tag{5.17}$$

Under the aforementioned two assumptions, the spatial distribution of the false alarms is uniform. Hence, the probability density function of a false measurement in the volume V, denoting the subspace in which the measurement is known to fall, is given by

$$p(z|z \text{ is a false measurement}) = \frac{1}{V} \tag{5.18}$$

The above model is used for random clutter only, whereas persistent clutter can be easily recognized and, then, eliminated.

Chapter 6

Data Association

This chapter addresses the data association problem in target tracking. Data association is fundamental in the presence of measurements whose origin is uncertain. This occurs when remote sensing devices acquire observations which can originate from true targets or clutter, and when multiple targets are present in the same surveillance region, especially if their trajectories are close to each other. The overall data association is the combination of the following different problems:

- track initialization (measurement-to-measurement association).
- measurement-to-track association (always preceded by a gating procedure).
- track-to-track association (for multisensor systems).

In this chapter track-to-track association is ignored, the focus being on track formation, measurement validation and, above all, measurement-to-track association. First, track initialization determines the presence of a target, in order to initiate the corresponding track and start the state estimation. Gating is a screening procedure, useful to determine which observations are better for updating existing tracks. Moreover, it is performed in order to avoid

unnecessary computations in the association procedure. In fact, association collects the measurement-to-track pairings that passed through the first gating, and establishes which are the best observation-to-track assignments to make. There are two basic different approaches in associating data.

- Non-Bayesian data association: statistical tools, such as maximum likelihood or hypothesis testing, are used in order to come to a binary (hard) decision. It is made for each measurement on whether it originates from a particular target or is a false alarm due to random clutter.
- Probabilistic or Bayesian data association: association probabilities are evaluated for each measurement and for all possible sources so as to take a soft decision. Subsequently, such association probabilities are used throughout the estimation process.

Data association approaches can also be categorized with respect to the way in which they process measurements, into

- single-scan: the estimation of the state of targets is based on the set of measurements at the current scan only.
- multi-scan: the association decisions take into account also measurements at previous scans.

6.1 Track Initialization

Track formation is a crucial issue in target tracking, especially because some data association techniques work under the assumption that the track has already been initialized. In these cases, track formation has to be implemented in order to start the tracking process. For radar tracking problems the process of track initialization is essentially carried out in two phases:

1. generation of the initial estimate using two observations;

2. track confirmation logic, used to reduce the amount of false tracks initialized.

Note that the first stage must assure the consistency of the estimation filter, i.e. the covariance of the initial estimate must reflect realistically its accuracy so that estimation errors are consistent with the evaluated covariances. In the next sections, first the simple and common two-point difference (TPD) track initialization algorithm will be first illustrated. Subsequently, the M/N logic, used for track confirmation, will be described.

6.1.1 Two-Point Differencing Gating

The TPD [3] algorithm approximates the initial track state using the first two position observations. Any consecutive pair of detections within a maximum distance, depending on the maximum speed and measurement noise variances of the target of interest, initiates a preliminary track. This preliminary track, consisting of the initial state and the corresponding covariance, can initialize the filter. In order to initialize the tracking filter for a target moving with nearly constant velocity when position-only measurements are available, consider two components of the state, position ξ and velocity $\dot{\xi}$ in a given coordinate. The measurement is modeled as

$$z(k) = \xi(k) + v(k) \tag{6.1}$$

so one generates the measurement noises for the true values $\xi(k), k = 1, 2$

$$v(k) \sim wn(0, R) \tag{6.2}$$

then position and velocity estimates at scan $k = 2$ can be obtained as follows

$$\begin{aligned} \hat{\xi}(2|2) &= z(2) \\ \hat{\dot{\xi}}(2|2) &= \frac{z(2) - z(1)}{T} \end{aligned} \tag{6.3}$$

and the corresponding initial covariance matrix, assuming there is no process noise, is

$$P(2|2) = \begin{bmatrix} R & R/T \\ R/T & 2R/T^2 \end{bmatrix} \tag{6.4}$$

This method guarantees consistency of the initialization of the tracking filter, that starts updating the state at time $k = 2$.

6.1.2 M/N Logic

Once a tentative track becomes a preliminary track, in order to reduce the amount of false tracks in a dense clutter environment, a M/N logic is implemented. This is a straightforward procedure used for track confirmation. The idea of this logic is that a preliminary track needs a minimum of M detections, during the first N scans ($1 \leq M < N$), to become a confirmed track. Tracks assume a different role throughout the confirmation and deletion processes.

- New tracks are not proper tracks, but simply consist of measurements rejected by the data association, for which a new TPD gate will be set up in the subsequent scan.
- Preliminary tracks are initiated by any consecutive pair of detections, within a maximum distance. Such tracks are created by the two-point differencing technique. A track remains preliminary until it is either confirmed or deleted, the decision relying on the M/N logic.
- Confirmed tracks are obtained from preliminary tracks after M out of N scans.
- Deleted tracks are either preliminary tracks that do not reach the M/N required detections or confirmed tracks that are terminated by the tracker.

The state diagram of the M/N logic is illustrated in figure 6.1

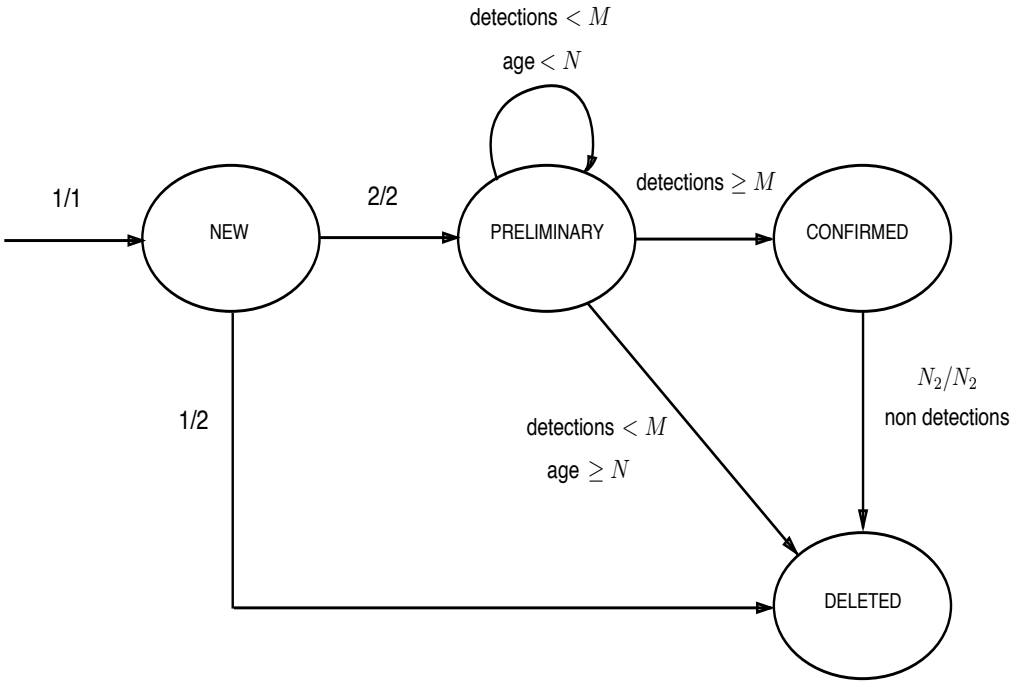

Figure 6.1: State diagram of M/N logic.

Note that a confirmed track is terminated if one of the following two conditions occurs:

- there are no validated measurements during N_2 consecutive scans, i.e. there are N_2 consecutive misdetections.
- the estimated target speed exceeds a maximum threshold.

Notice that not only confirmed tracks are processed in the filtering and in the data association blocks; preliminary tracks are processed as well. This is due to the necessity of updating these tracks in order to evaluate a validation region at subsequent scans and therefore continue the M/N procedure. This is possible only if the state predicted estimate and its covariance are available.

Although both preliminary and confirmed tracks are updated, only the latter will be displayed on the radar screen.

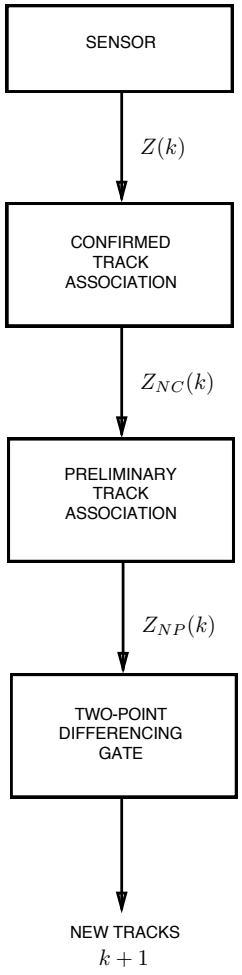

Figure 6.2: Data association priorities.

In data association there is a different priority given to new, preliminary and confirmed tracks, as shown in figure 6.2. As a matter of fact, the whole set of measurements $Z(k)$ provided by a sensor at time k is available only for the data association of confirmed tracks. The subset of measurements $Z_{NC}(k)$, rejected by the confirmed track association, will then be available for preliminary tracks. Lastly, the remaining subset of measurements $Z_{NP}(k)$,

discarded by both confirmed and preliminary tracks, will initiate new tracks at the following scan.

The track initialization procedure for multitarget tracking, implemented in the simulations described later, follows the logic illustrated in the block diagram of figure 6.3.

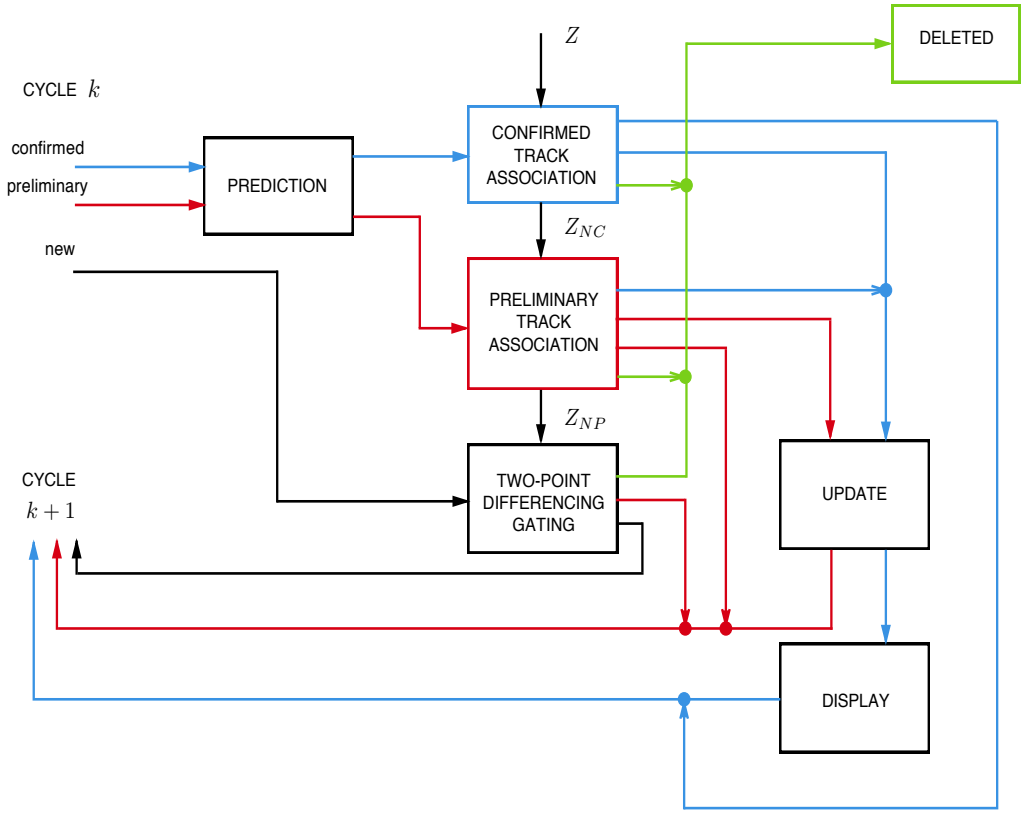

Figure 6.3: Block diagram of track initialization, maintenance and deletion throughout data association and filtering.

As shown in figure 6.3, at current time k, confirmed, preliminary and new tracks are available in the filter. The first two sets of tracks enter the prediction block, where the predicted state estimate, the corresponding covariance, the predicted measurement and its covariance are computed. Then, the set of

measurements are validated, through a measurement-to-track association, which at first involves only confirmed tracks. Subsequently, preliminary tracks validate measurements among the ones left in the subset Z_{NC}. The remaining Z_{NP} observations are collected in the TPD block, where they are possibly associated to new tracks found at scan $k-1$. After this three-level measurement-to-track association step, confirmed and preliminary tracks, whose measurements have been detected at the current time, are updated. Note that the data entering the association block as preliminary, and exiting the same block as confirmed, represents preliminary tracks that reach the M-th detection out of N, and hence are promoted to confirmed tracks. Only confirmed tracks are shown on the *Plan Position Indicator (PPI)* after the update. The preliminary tracks, at the end of the processing cycle at time k, will be the combination of the following tracks:

- updated preliminary tracks
- preliminary tracks which have been associated to no measurements, and thus their state estimates and covariances have only been predicted.
- tracks that have been created in the TPD block, after two detections in two consecutive scans

The new tracks, available at time $k+1$, will be given by the unassociated measurements at scan k. Deleted tracks, denoted by green lines in Figure 6.3, originate from confirmed, preliminary and new tracks for which termination conditions hold.

The choice of M and N depends on both probability of target detection P_D and probability of false alarm P_{FA}. As a matter of fact, for a given N, the product $P_D N$ represents the number of expected detections for a true track in N scans. Therefore, in order to confirm true tracks, a reasonable value

should provide

$$M < P_D N \tag{6.5}$$

On the other hand, the initialization logic should also prevent false tracks from being confirmed. This additional requirement suggests to choose M and N so that

$$A_{VR}\, P_{FA}\, N < M \tag{6.6}$$

where A_{VR} is the average area of the validation region. In conclusion, combining (6.5) and (6.6), the integers M and N should satisfy the following condition:

$$A_{VR} P_{FA} < \frac{M}{N} < P_D \tag{6.7}$$

The choice of N should also take into account the fact that only confirmed tracks are shown on the display. Thus the number of scans, necessary for track confirmation, cannot be too high.

6.2 Measurement Validation

In target tracking, measurements detected by sensors are first selected so that only the most likely to be target-originated are preserved for the subsequent data association. For each scan a validation gate is set up, so that only the observations which fall into that limited region will be valid candidates to update the existing tracks, while the remaining ones are assumed to be new targets or clutter. This selection ensures reduction of the computational burden and is mainly performed to avoid searching for observations in the entire measurement space. The validation region guarantees that the target observation falls in it with high probability, namely the gate probability P_G.

Once a track has been initialized, the filter can evaluate a predicted measurement expressed by its mean $\hat{z}(k|k-1)$ and its corresponding covariance $S(k)$.

Under the basic assumption that the measurement conditioned on the past is normally distributed with probability density function

$$p[z(k)|Z^{k-1}] = \mathcal{N}[z(k); \hat{z}(k|k-1), S(k)] \tag{6.8}$$

the true measurement will fall in the validation or gate region

$$\Upsilon(k,\gamma) = \{z : [z - \hat{z}(k|k-1)]' S(k)^{-1} [z - \hat{z}(k|k-1)] \leq \gamma\} \tag{6.9}$$

where γ is the gate threshold that defines the confidence region, chi-square distributed, with n_z (dimension of the measurement vector) degrees of freedom. $S(k)$ is the covariance of the innovation. The validation region, for Gaussian multivariate distributions, is the ellipsoid of minimum volume containing the probability mass P_G, centered at the mean, with semiaxes the square roots of the eigenvalues of the matrix γS. The gate probability is

$$P_G = Prob\{z(k) \in \Upsilon(k,\gamma)\} \tag{6.10}$$

The threshold γ can be evaluated given P_G and the dimension n_z from the table of the chi-square distribution. Sometimes, instead of γ, the number of sigmas (sigma is the standard deviation) of the gate $g = \sqrt{\gamma}$ is used. The volume of the validation region Υ corresponding to $\gamma = g^2$ (g-sigma gate) is

$$V(k) = c_{n_z} |\gamma S(k)|^{\frac{1}{2}} = c_{n_z} g^{n_z} |S(k)|^{\frac{1}{2}} \tag{6.11}$$

where c_{n_z} is the volume of the n_z-dimensional unit hypersphere, e.g. $c_2 = \pi$.

As a conclusion, the actual gating process accepts only observations whose statistical distance d^2 of the measurement-to-track assignment satisfies the following relationship:

$$d^2(k) = \nu(k)' S(k)^{-1} \nu(k) \leq \gamma \tag{6.12}$$

where $\nu(k)$ is the innovation vector, given by the difference between the actual and the expected measurement. The above distance is often referred as Normalized Innovation Squared (NIS), which is, as mentioned before, chi-square distributed if $\nu(k) \sim \mathcal{N}(0, S(k))$.

6.3 Single Target Data Association

When tracking a single target in clutter, one encounters a data association problem. This is due to the fact that there are possibly several measurements in the validation region of the target, discussed in Section 6.2. As a matter of fact, the validated measurements consist of:

- The target-originated measurement. This occurs if the correct observation is first detected and then validated.
- Clutter or false alarms

In this section, only single-scan data association approaches are discussed. First, the Nearest Neighbor (NN) method, which assigns to the target the closest observation to the predicted measurement, is revised. This approach, described in Section 6.3.1, is obviously non-Bayesian. An alternative is to use a Bayesian approach, called Probabilistic Data Association (PDA) and discussed in Section 6.3.2, which associates with a certain probability, each measurement to the target of interest.

6.3.1 Nearest Neighbor

The simplest approach to data association is the Nearest Neighbor (NN), which considers only one measurement-to-track hypothesis, choosing the most likely measurement for track update, according to a certain distance measure between the track and the measurement. The NN method is a typical single-scan algorithm, since it considers only the set of measurements received in the current scan to make a data association hard decision. Due to the fact that the Nearest Neighbor approach allows each track to be associated with the closest measurement, several tracks can have, as their nearest neighbor, the same single observation. Therefore, the same measurement may be used to update more than one track. This leads to measurement-to-track

misassignments which can cause the filter to converge slowly or even fail to converge. Consequently, the NN method is a straightforward algorithm whose implementation is recommended in a low-clutter environment. Furthermore, due to the aforementioned reasons, in multitarget tracking scenarios, the NN approach is not optimal, but another version called Global Nearest Neighbor (GNN), which prohibits multiple assignments of a single observation, can be used as described later.

6.3.2 Probabilistic Data Association

Probabilistic data association (PDA), first introduced by Bar-Shalom and Tse [1, 4], is a Bayesian approach to solve the problem of assigning measurements to tracks, significantly effective in dense clutter. This method associates validated measurements in the current scan with existing tracks by calculating the probability for each observation of being target-originated. Subsequently the association probabilities are used as weights for the track update. PDA is valid under the following assumptions

- A single target being tracked;
- The track has been already initialized;
- A validation region, calculated as in Section 6.2, is set up at each scan so that only a part of the set of measurements is considered for the association to the target;
- A single validated measurement can originate from the target;
- The other measurements are considered false alarms. They are modeled as independent and identically distributed (i.i.d.) with uniform spatial distribution, as shown in Section 5.5
- For each scan, the probability of target detection is P_D.

- The probability density function of the current state conditioned on the past is Gaussian, with mean given by the predicted state and covariance $P(k|k-1)$. Hence the past information about the target is summarized approximately by the following sufficient statistics

$$p[x(k)|Z^{k-1}] = \mathcal{N}[x(k); \hat{x}(k|k-1), P(k|k-1)]. \tag{6.13}$$

The set of $m(k)$ validated observations at the current time is

$$Z(k) = \{z_i(k)\}_{i=1}^{m(k)} \tag{6.14}$$

and the possible association events are

$$\theta_i(k) = \begin{cases} \{z_i(k) \text{ is the target-originated measurement}\} & i = 1, ..., m(k) \\ \{\text{all the measurements are false alarms}\} & i = 0 \end{cases} \tag{6.15}$$

As a consequence of the aforementioned assumptions, for $m(k) \geq 1$, these events turn out to be mutually exclusive and collectively exhaustive i.e. they cannot occur together, but at least one of the events must happen so that their probabilities sum up to unity. The association probability corresponds to the conditional probability of the i-th association event being the correct one, defined as

$$\beta_i(k) = Prob\{\theta_i(k)|Z^k\} \tag{6.16}$$

where

$$Z^k = \{Z(j)\}_{j=1}^{k} \tag{6.17}$$

is the cumulative sequence of observations up to time k. The conditioning in (6.16) can be divided into the past set of measurements Z^{k-1} and the current one $Z(k)$. In particular, considering that a Bayesian inference can be used on both the number of validated observations $m(k)$, and on their set of values $Z(k)$, (6.16) becomes

$$\beta_i(k) = Prob\{\theta_i(k)|Z(k), m(k), Z^{k-1}\} \tag{6.18}$$

The application of Bayes' theorem leads to the explicit formula

$$\beta_i(k) = \frac{1}{c}\, p\left[Z(k)|\theta_i(k), m(k), Z^{k-1}\right] Prob\left\{\theta_i(k)|m(k), Z^{k-1}\right\} \quad i = 0, 1, ..., m(k) \tag{6.19}$$

where: c is a normalization constant; $Prob\left\{\theta_i(k)|m(k), Z^{k-1}\right\}$ is the prior probability of measurement i being the correct one; $p\left[Z(k)|\theta_i(k), m(k), Z^{k-1}\right]$ is the joint density of the validated observations conditioned on measurement i being target-originated. Specifically, $p\left[Z(k)|\theta_i(k), m(k), Z^{k-1}\right]$ is the product of the assumed Gaussian pdf of the correct measurement $p\left[z_i(k)|\theta_i(k), m(k), Z^{k-1}\right]$ and the pdf $[V(k)]^{-1}$ of the false measurements, which is assumed uniform in the validation region volume. In particular, the pdf of the correct observation is

$$p\left[z_i(k)|\theta_i(k), m(k), Z^{k-1}\right] = P_G^{-1}\mathcal{N}[z_i(k); \hat{z}(k|k-1), S(k)] = P_G^{-1}\mathcal{N}[\nu_i(k); 0, S(k)] \tag{6.20}$$

i.e. it is Gaussian with mean equal to the predicted measurement $\hat{z}(k|k-1)$ and the innovation covariance $S(k)$ limited to the gate (indeed it is divided by the gate probability P_G, which is the probability that the correct observation is inside the validation region). Thus, the pdf from (6.19) is

$$p\left[Z(k)|\theta_i(k), m(k), Z^{k-1}\right] = \begin{cases} V(k)^{-(m(k)-1)} P_G^{-1}\mathcal{N}[\nu_i(k); 0, S(k)] & i = 1, ..., m(k) \\ V(k)^{-m(k)} & i = 0 \end{cases} \tag{6.21}$$

where $V(k)$, defined in (6.11), is the volume of the validation region. The probabilities of the association events conditioned only on the number of validated measurements are denoted as

$$\gamma_i[m(k)] \triangleq Prob\left\{\theta_i(k)|m(k), Z^{k-1}\right\} = Prob\left\{\theta_i(k)|m(k)\right\}$$

The parametric approach uses a Poisson model for the probability mass function (pmf) of the number of false measurements in the validation region,

with spatial density λ, so that

$$\mu_F(m) = e^{-\lambda V} \frac{(\lambda V)^m}{m!} \tag{6.22}$$

It can been demonstrated that

$$\gamma_i[m(k)] = \begin{cases} P_D P_G \left[P_D P_G m(k) + (1 - P_D P_G)\lambda V(k)\right]^{-1} & i = 1, ..., m(k) \\ (1 - P_D P_G)\lambda V(k) \left[P_D P_G m(k) + (1 - P_D P_G)\lambda V(k)\right]^{-1} & i = 0 \end{cases} \tag{6.23}$$

Then, the application of (6.21) and (6.23) to the pdf in (6.19) yields the following final expressions for the association probabilities:

$$\beta_i(k) = \begin{cases} \frac{1}{c} V(k)^{-(m(k)-1)} P_G^{-1} \mathcal{N}[\nu_i(k); 0, S(k)] P_D P_G & i = 1, ..., m(k) \\ \frac{1}{c} V(k)^{-m(k)} (1 - P_D P_G)\lambda V(k) & i = 0 \end{cases} \tag{6.24}$$

where the common terms of (6.23) $\left[P_D P_G m(k) + (1 - P_D P_G)\lambda V(k)\right]^{-1}$ are inside c, which is given by

$$c = \sum_{i=0}^{m(k)} \beta_i(k) \tag{6.25}$$

After some cancellations and dividing by P_D, the following expression is obtained

$$\beta_i(k) = \begin{cases} \frac{1}{c} \mathcal{N}[\nu_i(k); 0, S(k)] & i = 1, ..., m(k) \\ \frac{1}{c} \lambda \frac{(1 - P_D P_G)}{P_D} & i = 0 \end{cases} \tag{6.26}$$

that can be finally rewritten as

$$\beta_i(k) = \begin{cases} \dfrac{e_i}{1 - P_D P_G + \sum_{j=1}^{m(k)} e_j}, & i = 1, ..., m(k) \\ \dfrac{1 - P_D P_G}{1 - P_D P_G + \sum_{j=1}^{m(k)} e_j}, & i = 0 \end{cases} \tag{6.27}$$

where

$$e_i \triangleq \frac{\mathcal{N}\left[z_i(k); \hat{z}(k|k-1), S(k)\right] P_D}{\lambda} \tag{6.28}$$

is the likelihood ratio of the measurement $z_i(k)$ originating from the target rather than from clutter, i.e. the ratio of the pdf of the measurement $z_i(k)$ being the correct one over the pdf of being clutter, divided by the sum of all likelihood ratios. Note that λ in (6.28) plays the role of the uniform pdf of the location of a false measurement. Moreover, $1 - P_D P_G$ is the probability that no measurement is correct ($i = 0$). The innovation Gaussian in (6.27) is

$$\mathcal{N}\left[z_i(k); \hat{z}(k|k-1), S(k)\right] = \frac{1}{|2\pi S(k)|^{\frac{1}{2}}} e^{-\frac{1}{2}\nu_i(k)' S(k)^{-1} \nu_i(k)} \tag{6.29}$$

The association probabilities can be used in the tracking algorithm known as Probabilistic Data Association Filter (PDAF). If the state and measurement equations are linear it is based on the Kalman Filter, otherwise it uses EKF (Section 3.1) or UKF (Section 3.2). The prediction of the state, its covariance and the measurement is identical to the one in the standard Kalman Filter. For this reason, the next section is focused on the state and covariance update. The conditional mean of the state at scan k is given by

$$\hat{x}(k|k) = E[x(k)|Z^k] \quad \text{where } Z^k = \{z(i), i \leq k\} \tag{6.30}$$

Due to the association events being mutually exclusive and collectively exhaustive, the total probability theorem can be used with respect to $\theta_i(k)$, thus providing:

$$\begin{aligned} \hat{x}(k|k) &= Prob\{\theta_i(k)|Z^k\} \sum_{i=0}^{m(k)} E[x(k)|\theta_i(k), Z^k] \\ &= \beta_i(k) \sum_{i=0}^{m(k)} \hat{x}_i(k|k) \end{aligned} \tag{6.31}$$

where $\hat{x}_i(k|k)$ is the updated state conditioned on the event $\{z_i(k)$ is the correct (target- originated) validated measurement$\}$ that can be written as

$$\hat{x}_i(k|k) = \hat{x}(k|k-1) + K(k)\nu_i(k) \quad i = 1, ..., m(k) \tag{6.32}$$

where the innovation for the i-th measurement $\nu_i(k)$ is

$$\nu_i(k) = z_i(k) - \hat{z}(k|k-1) \tag{6.33}$$

and the gain is the same as the one in the standard Kalman Filter

$$K(k) = P(k|k-1)H(k)'S(k)^{-1} \tag{6.34}$$

Hence the state update equation of PDAF can be obtained by combining (6.32) into (6.31)

$$\hat{x}(k|k) = \hat{x}(k|k-1) + K(k)\nu(k) \tag{6.35}$$

where $\nu(k)$ is the combined innovation defined as follows:

$$\nu(k) = \sum_{i=1}^{m(k)} \beta_i(k)\nu_i(k) \tag{6.36}$$

Notice that, for $i = 0$ there is no contribution to the combined innovation, because either no measurement is inside the gate or the validated ones are clutter, i.e.

$$\hat{x}_0(k|k) = \hat{x}(k|k-1) \tag{6.37}$$

Similarly the covariance update is

$$\begin{aligned} P(k|k) &= E[(x(k) - \hat{x}(k|k))(x(k) - \hat{x}(k|k))'|Z^k] \\ &= \sum_{i=0}^{m(k)} E[(x(k) - \hat{x}(k|k))(x(k) - \hat{x}(k|k))'|\theta_i(k), Z^k]\beta_i(k) \\ &= \bar{P}(k|k) + \tilde{P}(k) \end{aligned} \tag{6.38}$$

where

$$\begin{aligned} \bar{P}(k|k) &= \sum_{i=0}^{m(k)} \beta_i(k)P_i(k|k) \\ \tilde{P}(k) &= \sum_{i=0}^{m(k)} \beta_i(k)\hat{x}_i(k|k)\hat{x}_i(k|k)' - \hat{x}(k|k)\hat{x}(k|k)' \end{aligned} \tag{6.39}$$

Notice that, for $i = 0$,

$$P_0(k|k) = P(k|k-1) \tag{6.40}$$

whereas, for $i \neq 0$, the covariance of the state updated with the correct measurement is

$$P_i(k|k) = P^c(k|k) = P(k|k-1) - K(k)S(k)K(k)' \qquad i = 1, ..., m(k) \quad (6.41)$$

Therefore

$$\bar{P}(k|k) = \beta_0(k)P(k|k-1) + [1 - \beta_0(k)]P^c(k|k) \quad (6.42)$$

where we have used the identity

$$\beta_0(k) = 1 - \sum_{i=0}^{m(k)} \beta_i(k) \quad (6.43)$$

The spread of the means $\tilde{P}(k|k)$ in (6.39) can be written as

$$\begin{aligned} \tilde{P}(k) &= \sum_{i=0}^{m(k)} \beta_i(k)\hat{x}_i(k|k)\hat{x}_i(k|k)' - \hat{x}(k|k)\hat{x}(k|k)' \\ &= \sum_{i=0}^{m(k)} \beta_i(k)[\hat{x}(k|k-1) + K(k)\nu_i(k)][\hat{x}(k|k-1) + K(k)\nu_i(k)]' \\ &- [\hat{x}(k|k-1) + K(k)\nu(k)][\hat{x}(k|k-1) + K(k)\nu(k)]' \end{aligned} \quad (6.44)$$

Using

$$\sum_{i=0}^{m(k)} \beta_i(k) = 1 \quad (6.45)$$

and

$$\sum_{i=0}^{m(k)} \beta_i(k)\nu_i(k) = \sum_{i=1}^{m(k)} \beta_i(k)\nu_i(k) = \nu(k), \quad (6.46)$$

since by definition

$$\nu_0(k) = 0, \quad (6.47)$$

(6.44) becomes

$$\begin{aligned} \tilde{P}(k) =& \hat{x}(k|k-1)\hat{x}(k|k-1)' + \hat{x}(k|k-1)[K(k)\nu(k)]' + K(k)\nu(k)\hat{x}(k|k-1)' \\ &+ \sum_{i=0}^{m(k)} \beta_i(k)K(k)\nu_i(k)[K(k)\nu_i(k)]' - \hat{x}(k|k-1)\hat{x}(k|k-1)' \\ &- \hat{x}(k|k-1)[K(k)\nu(k)]' - [K(k)\nu(k)]\hat{x}(k|k-1)' - K(k)\nu(k)[K(k)\nu(k)]' \end{aligned} \quad (6.48)$$

and after cancellations

$$\tilde{P}(k) = K(k)\big[\sum_{i=0}^{m(k)} \beta_i(k)\nu_i(k)\nu_i(k)' - \nu(k)\nu(k)'\big]K(k)' \tag{6.49}$$

This term, the spread of the innovations, accounts for the uncertainty caused by the unknown origin of the measurements. As a consequence, it is a positive semidefinite matrix which increases the covariance of the updated state that lastly becomes

$$P(k|k) = \beta_0(k)P(k|k-1) + [1-\beta_0(k)]P^c(k|k) + \tilde{P}(k) \tag{6.50}$$

From the above expression, it is clear that with probability $\beta_0(k)$ the covariance is the prediction one and there is no update because none of the observations are target originated. On the other hand $1-\beta_0(k)$ is the weight of the updated covariance $P^c(k|k)$ when the correct measurement is available.
In Algorithm 4 the pseudo-code summarizes the PDAF. In Figure 6.4 one single cycle of the filter is shown.

Algorithm 4 PDAF

function PDAF($\hat{x}(0|-1)$, $P(0|-1)$)

  for all time $k = 0, 1, 2, \ldots$ **do**

    Correction

    $S(k) \longleftarrow R(k) + C(k)P(k|k-1)C(k)'$

    $K(k) \longleftarrow P(k|k-1)C(k)'S(k)^{-1}$

    $\hat{z}(k|k-1) \longleftarrow C(k)\hat{x}(k|k-1)$

    for all measurements $z_i = 0, 1, 2, \ldots, m(k)$ **do**

      $\nu_i(k) \longleftarrow z_i(k) - \hat{z}(k|k-1)$

      $$\beta_i(k) = \begin{cases} \dfrac{e_i}{1 - P_D P_G + \sum_{j=1}^{m(k)} e_j}, & i = 1, ..., m(k) \\ \dfrac{1 - P_D P_G}{1 - P_D P_G + \sum_{j=1}^{m(k)} e_j}, & i = 0 \end{cases}$$

    end for

    $$\nu(k) \longleftarrow \sum_{i=1}^{m(k)} \beta_i(k)\nu_i(k)$$

    $\hat{x}(k|k) \longleftarrow \hat{x}(k|k-1) + K(k)\nu(k)$

    $P^c(k|k) \longleftarrow P(k|k-1) - K(k)S(k)K(k)'$

    $$\tilde{P}(k) \longleftarrow K(k)\Big[\sum_{i=0}^{m(k)} \beta_i(k)\nu_i(k)\nu_i(k)' - \nu(k)\nu(k)'\Big]K(k)'$$

    $P(k|k) \longleftarrow \beta_0(k)P(k|k-1) + [1 - \beta_0(k)]P^c(k|k) + \tilde{P}(k)$

    Prediction

    $\hat{x}(k+1|k) \longleftarrow A(k)\hat{x}(k|k)$

    $P(k+1|k) \longleftarrow A(k)P(k|k)A'(k) + D(k)Q(k)D(k)'$

  end for

  return prediction and correction sequence of $[\hat{x}, P]$

end function

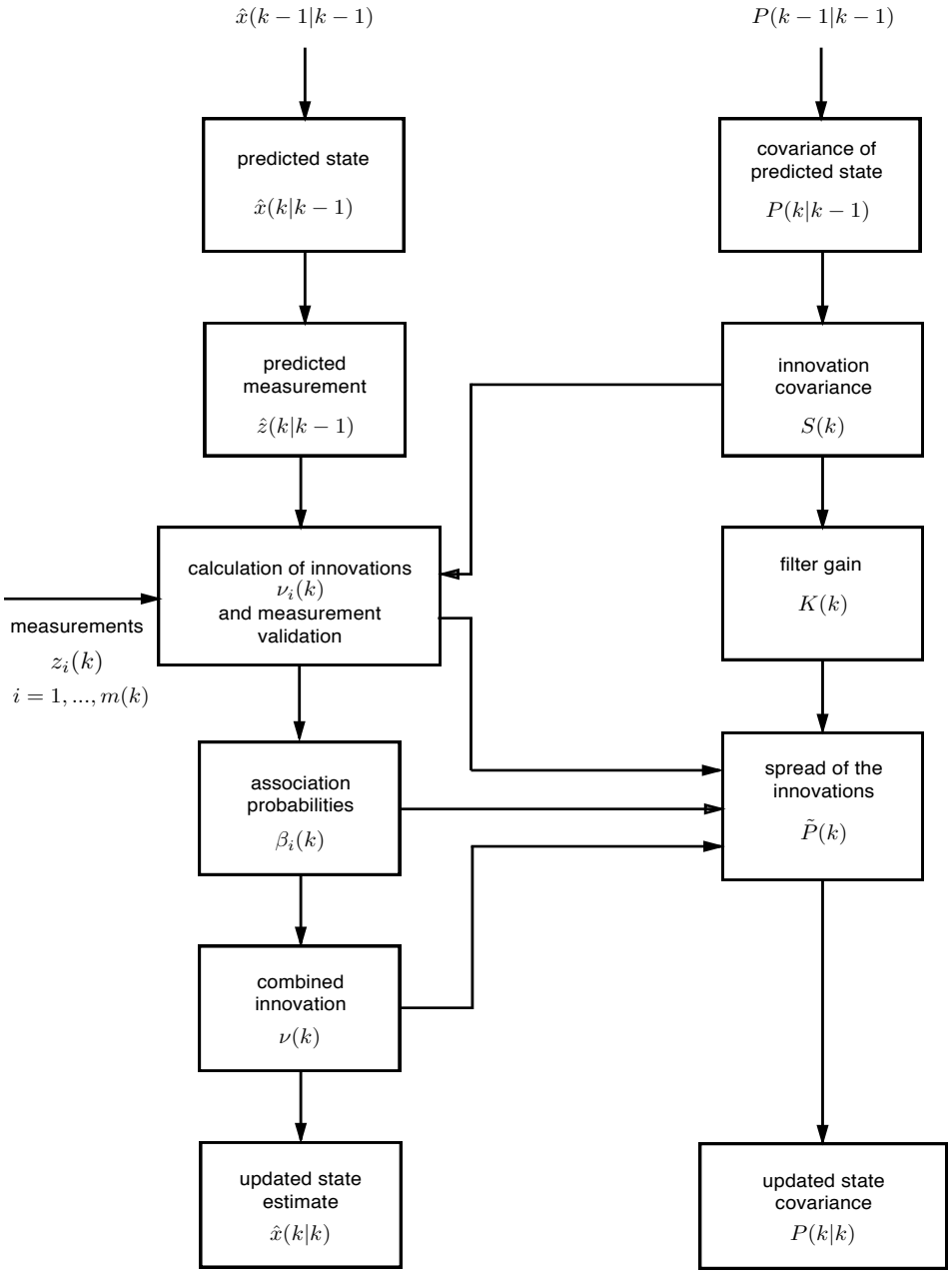

Figure 6.4: One cycle of the PDAF.

6.4 Multitarget Data Association

Data association in a multiple target scenario is more complicated. Several targets as well as clutter create uncertainty on the origin of measurements. In Figure 6.5 a multitarget situation is illustrated at a given time instant. The predicted measurements for the two targets are denoted as $\hat{z}_1$ and $\hat{z}_2$. The measurement z_1 can be originated from target 1 or clutter, z_3 and z_4 from target 2 or clutter, while z_2 from either target 1 or target 2 or clutter. When a measurement may have originated from more than one target (e.g. measurement z_2), it will be shown that these targets cannot be considered separately in the association problem.

This section focuses on Bayesian approaches in a multitarget environment, in which the association of measurements is performed by considering simultaneously all the targets. Nevertheless, a hard decision approach to this problem (Global Nearest Neighbor), is first discussed.

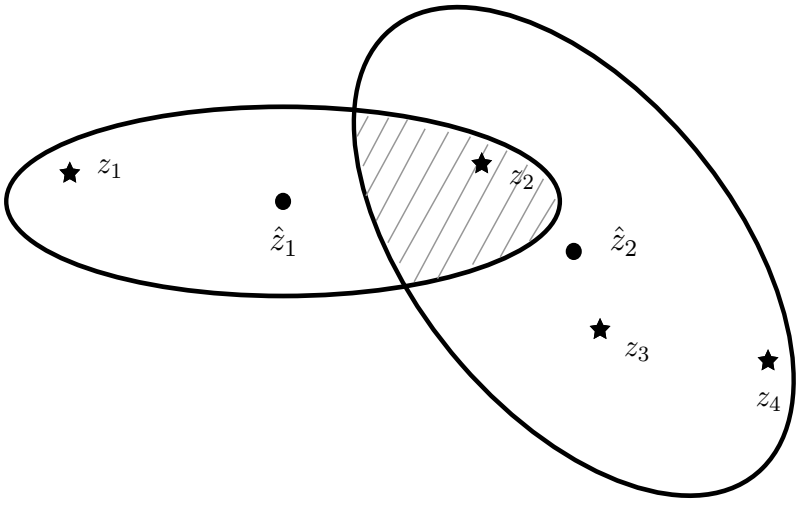

Figure 6.5: A measurement falls in the intersection of two validation regions.

6.4.1 Global Nearest Neighbor

Global Nearest Neighbor (GNN) is an extension of the NN approach to the multitarget case, with the difference that each validated measurement can be assigned only to a single track. The association problem is formulated as the well-known 2-D assignment problem, which makes a hard decision on the measurement-to-track association and can hence be solved by algorithms commonly used for this purpose, such as the auction or the Hungarian methods. These algorithms provide optimal and computationally less demanding suboptimal solutions to the assignment problem.

Let N_T be the number of tracks and m the number of measurements. The classical assignment problem consists in finding a one-to-one matching between N_T tracks and m measurements. The objective is to minimize the total cost of the assignments. The mathematical formulation for the assignment problem can be given in terms of the following discrete constrained optimization problem:

$$\text{Minimize} \qquad \sum_{j=1}^{N_T}\sum_{l=1}^{m} c_{jl}a_{jl} \tag{6.51}$$

$$\text{Subject to :} \qquad \begin{aligned} &\sum_{j=1}^{N_T} a_{jl} \leq 1 \qquad l = 1, \dots m, \\ &\sum_{l=1}^{m} a_{jl} \leq 1 \qquad j = 1, \dots N_T, \end{aligned} \tag{6.52}$$

$$a_{jl} \in 0, 1 \tag{6.53}$$

where $a_{jl} = 1$ if track j is assigned to measurement l, $a_{jl} = 0$ otherwise. The quantity c_{jl} is the cost of assigning observation l to track j. The first set of constraints ensures that each measurement can be assigned at most to a single track. The second set of constraints, on the other hand, imposes that each track is assigned to at most a single measurement. An optimal solution to this problem is described in Section 8.2.3.

In the GNN problem, for each possible assignment of a measurement l to a track j, a Mahalanobis distance d_{jl} is defined by:

$$d_{jl} = \big[z_l(k) - \hat{z}_j(k|k-1)\big]' S_j(k)^{-1} \big[z_l(k) - \hat{z}_j(k|k-1)\big] \tag{6.54}$$

As mentioned in Section 6.3.1, the NN assigns to each track the closest measurement, according to (6.54). On the other hand, the GNN algorithm solves the assignment problem via minimization of the cost c_{jl} in (6.51), which is related to the distance in (6.54) as follows

$$c_{jl} = -\log(d_{jl}/P_{FA}) \tag{6.55}$$

where P_{FA} is the probability of false alarm. In conclusion, the Global Nearest Neighbor technique seeks for the globally optimal solution with respect to the measurement-to-track assignment costs. A practical comparison between NN and GNN assignment decisions is shown in Figure 6.6. As illustrated in the figure, GNN, unlike NN, finds the optimal solution to the assignment problem, preventing associations of a measurement to multiple tracks.

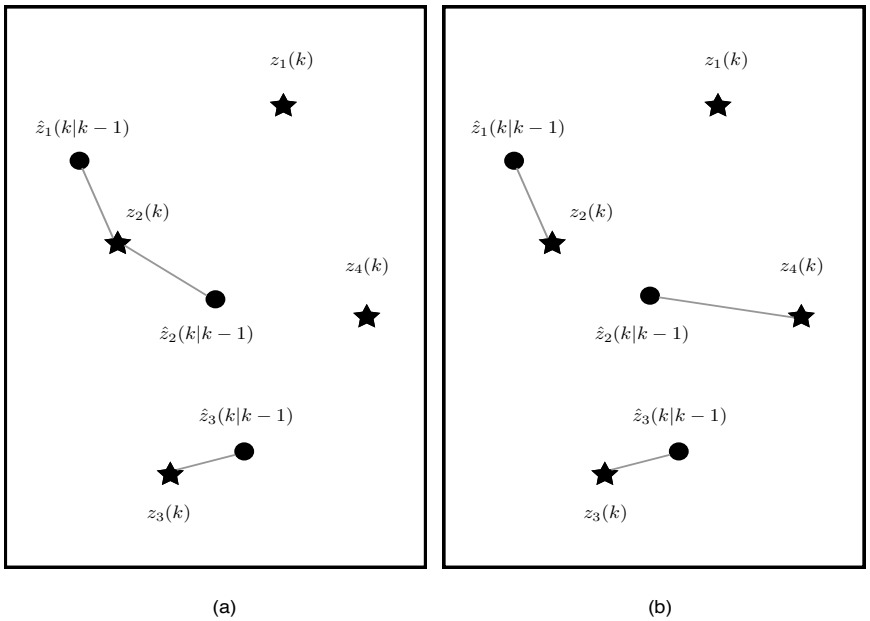

Figure 6.6: Data association in NN (a) and GNN (b).

6.4.2 Joint Probabilistic Data Association

Joint Probabilistic Data Association [1, 2] is the direct extension, for multiple target scenarios in clutter, of PDA introduced in section 6.3.2. Specifically, JPDA takes also into account the possibility that a measurement inside the validation region of a certain track, may originate from a different target, that has the same measurement in its gate. As a consequence, the basic difference with PDA is that the measurement-to-track association probabilities are calculated jointly across the targets, so that every measurement falling inside the surveillance region is used for the evaluation of the association probabilities. On the other hand, in the PDA filter the weights are calculated independently track by track. For each track, PDA computes the association probability of each measurement in the validation region of the track, using the Bayes rule. PDA does not consider any other observation in the track gate to be originated from another target. Conversely, the probability calculations of the JPDA filter do consider other measurements coming from other targets. For this reason, JPDA computations are quite complex and computationally intensive. Note that, if there is no intersection between validation regions of different tracks, JPDA and PDA are equivalent. Differences arise when trajectories of multiple targets are close to each other. First, the probability of all possible feasible joint events is calculated. A feasible joint event is a non-conflicting association of current tracks with measurements, based on the requirement that two tracks cannot be associated to the same track. The remaining unassociated observations are assumed clutter points. JPDA evaluates the conditional probabilities of the joint association events at the current time as follows:

$$\theta(k) = \bigcap_{j=1}^{m} \theta_{ji_j}(k) \tag{6.56}$$

where $\theta_{ji}(k)$ is the event that at time k measurement j originated from target i, $j = 1, \ldots, m$, and $i = 0, 1, \ldots, N_T$. i_j denotes that measurement j is

associated to target i in the specific event, and N_T is the known number of established targets. Validation gates are only used to select the feasible joint events, i.e. events whose probabilities cannot be neglected and hence affect the other probabilities. In particular, a joint association event θ is represented by its event matrix

$$\hat{\Omega}(\theta) = [\hat{\omega}_{ji}(\theta)] \tag{6.57}$$

where

$$\hat{\omega}_{ji}(\theta) = \begin{cases} 1, & \text{if } \theta_{ji} \in \theta \\ 0, & \text{otherwise} \end{cases} \tag{6.58}$$

For a feasible association event, the following conditions are accomplished:

- a measurement can have only one source, either a target or clutter, i.e.

$$\sum_{i=0}^{N_T} \hat{\omega}_{ji}(\theta) = 1 \qquad j = 1, \ldots, m \tag{6.59}$$

- at most a single measurement can originate from a target i.e.

$$\delta_i(\theta) = \sum_{j=1}^{m} \hat{\omega}_{ji}(\theta) \leq 1 \qquad i = 1, \ldots, N_T \tag{6.60}$$

where $\delta_i(\theta)$ is the target detection indicator. The variable which indicates if measurement j is associated with a target in event θ is the measurement association indicator:

$$\tau_j(\theta) \triangleq \sum_{i=1}^{N_T} \hat{\omega}_{ji}(\theta) \tag{6.61}$$

Thus, the number of false measurements in event θ is

$$\phi(\theta) = \sum_{j=1}^{m} [1 - \tau_j(\theta)] \tag{6.62}$$

In order to evaluate the joint probabilities, consider the probability of an event conditioned on the past measurements Z^k up to time k

$$Prob\{\theta(k)|Z^k\} = Prob\{\theta(k)|Z(k), m(k), Z^{k-1}\} \tag{6.63}$$

the application of Bayes' formula yields

$$\begin{aligned} Prob\{\theta(k)|Z^k\} &= \frac{1}{c}\, p\left[Z(k)|\theta(k), m(k), Z^{k-1}\right] Prob\{\theta(k)|Z^{k-1}, m(k)\} \\ &= \frac{1}{c}\, p\left[Z(k)|\theta(k), m(k), Z^{k-1}\right] Prob\{\theta(k)|m(k)\} \end{aligned} \tag{6.64}$$

where c is a normalization constant. Under the assumption that the states of the targets conditioned on the past observations are mutually independent, the likelihood function of the measurements is

$$p\left[Z(k)|\theta(k), m(k), Z^{k-1}\right] = \prod_{j=1}^{m(k)} p\left[z_j(k)|\theta_{ji_j}(k), Z^{k-1}\right] \tag{6.65}$$

where i_j is the index of the target to which measurement j is associated in the event under consideration; since $m(k)$ are the measurements in the union of the validation regions at time k, the above product form is used.
The conditional pdf of a measurement given its origin is

$$p\left[z_j(k)|\theta_{ji_j}(k), Z^{k-1}\right] = \begin{cases} f_{i_j}\left[z_j(k)\right], & \text{if } \tau_j\left[\theta(k)\right] = 1 \\ V^{-1}, & \text{if } \tau_j\left[\theta(k)\right] = 0 \end{cases} \tag{6.66}$$

where

$$f_{i_j}\left[z_j(k)\right] = \mathcal{N}\left[z_j(k); \hat{z}^{i_j}(k|k-1), S^{i_j}(k)\right] \tag{6.67}$$

and $\hat{z}^{i_j}(k|k-1)$ is the predicted measurement for target i_j, with corresponding innovation covariance $S^{i_j}(k)$. The presence of V^{-1} in (6.66) is due to the assumption that unassociated measurements are uniformly distributed over the surveillance region of volume V. Hence, the pdf in (6.65) can be rewritten as

$$p\left[Z(k)|\theta(k), m(k), Z^{k-1}\right] = V^{-\phi} \prod_j \{f_{t_j}\left[z_j(k)\right]\}^{\tau_j} \tag{6.68}$$

where $\phi(\theta)$ is the total number of false measurements in the event $\theta(k)$ and τ_j are the indicators that select the single measurement densities according to their associations in event $\theta(k)$.

The last term to be determined in (6.64) is the prior probability

$$Prob\{\theta(k)|m(k)\} = Prob\{\theta(k), \delta(\theta), \phi(\theta)|m(k)\} \tag{6.69}$$

where $\delta(\theta)$ is the vector of target detection indicators corresponding to event $\theta(k)$. Note that the above expression follows from the fact that, given θ, both $\delta(\theta)$ and the number of false measurements $\phi(\theta)$ are known. Then, the joint probability can be rewritten as

$$Prob\{\theta(k)|m(k)\} = Prob\{\theta(k)|\delta(\theta), \phi(\theta), m(k)\}Prob\{\delta(\theta), \phi(\theta)|m(k)\} \tag{6.70}$$

The factor follows from a combinatorial reasoning:

- The number of targets assumed detected in event $\theta(k)$ is $m(k) - \phi$
- The set of measurement-to-target assignment events $\theta(k)$ in which the same number of targets is detected, can be computed as the permutations of the $m(k)$ measurements taken as $m(k) - \phi$.

Hence, if each association is equally likely i.e.

$$Prob\{\theta(k)|\delta(\theta), \phi(\theta), m(k)\} = \left(\frac{m(k)!}{\phi!}\right)^{-1} \tag{6.71}$$

assuming δ and ϕ independent, the second factor in (6.70) can be written as

$$Prob\{\delta(\theta), \phi(\theta)|m(k)\} = \prod_i (P_D^i)^{\delta_i}(1 - P_D^i)^{1-\delta_i}\mu_F(\phi) \tag{6.72}$$

where P_D^i is the detection probability of target i, $\mu_F(\phi)$ is the probability mass function of the number of false measurements and δ_i indicates that a measurement has been assigned to track i in the joint event $\theta(k)$. Combining (6.71) and (6.72) into the joint probability (6.70), the prior probability of a joint association event $\theta(k)$ is obtained as

$$Prob\{\theta(k)|m(k)\} = \frac{\phi!}{m(k)!}\mu_F(\phi)\prod_i (P_D^i)^{\delta_i}(1 - P_D^i)^{1-\delta_i} \tag{6.73}$$

Finally, combining (6.68) and (6.73) into (6.64), the posterior probability of a joint association event $\theta(k)$ turns out to be:

$$Prob\{\theta(k)|Z^k\} = \frac{1}{c}\frac{\phi!}{m(k)!}\mu_F(\phi)V^{-\phi}\prod_j\{f_{i_j}[z_j(k)]\}^{\tau_j}\prod_i(P_D^i)^{\delta_i}(1-P_D^i)^{1-\delta_i} \tag{6.74}$$

where ϕ, δ_i and τ_j depend on the considered event $\theta(k)$.
Assuming that the probability mass function of the number of false measurements $\mu_F(\phi)$ is a Poisson distribution, one has

$$\mu_F(\phi) = e^{-\lambda V}\frac{(\lambda V)^\phi}{\phi!} \tag{6.75}$$

where λ is the spatial density of false measurements. Using (6.75) in (6.74), $V^{-\phi}$ and $\phi!$ cancel. Moreover, each term contains $e^{-\lambda V}$ and $m(k)!$, which also cancel, since they appear in the denominator c of (6.74), which is the sum of all the numerators. The joint association probabilities of the parametric JPDA are, therefore,

$$Prob\{\theta(k)|Z^k\} = \frac{\lambda^\phi}{c_1}\prod_j\{f_{i_j}[z_j(k)]\}^{\tau_j}\prod_i(P_D^i)^{\delta_i}(1-P_D^i)^{1-\delta_i} \tag{6.76}$$

where c_1 is the appropriate normalization constant. Since $m(k)$ is a fixed number, a new normalization constant can be defined as

$$c_2 \triangleq c_1\lambda^{-m(k)} \tag{6.77}$$

Using c_2 in (6.76), the parametric joint association probabilities can be finally rewritten as

$$Prob\{\theta(k)|Z^k\} = \frac{1}{c_2}\prod_j\{\lambda^{-1}f_{i_j}[z_j(k)]\}^{\tau_j}\prod_i(P_D^i)^{\delta_i}(1-P_D^i)^{1-\delta_i} \tag{6.78}$$

After computing the joint probabilities (6.78), the association probability $\beta_{ji}(k)$ of track i to measurement j at scan k, is the sum of the probabilities of all joint events in which the marginal event of interest occurs. According

to the law of total probability:

$$\beta_{ji} \triangleq Prob\{\theta_{ji}|Z^k\} = \sum_{\theta} Prob\{\theta|Z^k\}\,\hat{\omega}_{ji}(\theta) \tag{6.79}$$

where $\hat{\omega}_{ji}(\theta)$ was defined in (6.58). The state estimation equations are exactly the same as in the PDAF, discussed in Section 6.3.2

6.4.3 Cheap Joint Probabilistic Data Association

Since every possible hypothesis of association between measurements and existing tracks has to be considered, the computation of the association probabilities in the standard JPDA approach becomes quite complicated. For this reason, in [14] Fitzgerald proposed a computationally *cheap* version of the JPDA filter, called Cheap Joint Probabilistic Data Association (CJPDA). The basic idea of this method is to provide an approximation of the association probabilities to alleviate the computational burden. At the same time, this technique aims to limit association performance degradation.

The CJPDA formula calculates the probability of track j being associated with measurement l as

$$\beta_{lj} = \frac{G_{lj}}{\sum_{i=1}^{N_T} G_{li} + \sum_{i=1}^{m} G_{ij} - G_{lj} + B} \tag{6.80}$$

where G_{lj} is proportional to the Gaussian likelihood function and indicates the closeness of fit of track j with the l-th measurement. It is given by:

$$G_{lj} = \frac{1}{|S_{lj}|^{1/2}} e^{-\frac{1}{2}\nu'_{lj} S_{lj}^{-1} \nu_{lj}} \tag{6.81}$$

where S_{lj} is the covariance matrix of the innovations ν_{lj}. B is a constant which depends on clutter density and detection probability, it can be set to zero when the clutter is not significant. Specifically, the parameter B can be computed as follows

$$B = \frac{P_{FA}}{A_{RC}} \frac{1 - P_D P_G}{P_G} \tag{6.82}$$

where A_{RC} is the area of the radar cell and P_{FA} is the false alarm probability. Note that the formula (6.80) reduces the association probability when there are overlapping gates and a measurement falls inside regions of different tracks. In this case, the weight is lowered by a large $\sum_{i=1}^{N_T} G_{li}$. In addition, if the track has several measurements to choose from, also $\sum_{i=1}^{m} G_{ij}$ will be large and the probability will be smaller. In conclusion, this calculation gives higher weights to those measurements closer to the predicted position and which are validated by the a smaller number of tracks. The expression reduces to the correct form when there is only a single target and several observations.

There are few useful remarks to this formulation. First, Fitzgerald states that only the three measurements with the highest probabilities for each track should be considered in the subsequent combined innovation. This accounts for processor loading considerations. Furthermore, probabilities below a given threshold can be set to zero, so that low-probability updates are prevented.

Algorithm 5 CJPDAF

function CJPDAF($\hat{x}(0|-1)$, $P(0|-1)$)

  for all time $k = 0, 1, 2, \ldots$ **do**

    for all tracks $j = 0, 1, 2, \ldots, N_T$ **do**

    Correction

      $S_j(k) \longleftarrow R(k) + C(k)P_j(k|k-1)C(k)'$

      $K_j(k) \longleftarrow P_j(k|k-1)C(k)'S_j(k)^{-1}$

      $\hat{z}_j(k|k-1) \longleftarrow C(k)\hat{x}_j(k|k-1)$

      for all measurements validated overall $z_l \in m(k)$ **do**

        $\nu_{lj}(k) \longleftarrow z_l(k) - \hat{z}_j(k|k-1)$

        $$\beta_{lj}(k) = \frac{G_{lj}(k)}{\sum_{i=1}^{N_T} G_{li}(k) + \sum_{i=1}^{m(k)} G_{ij}(k) - G_{lj}(k) + B}$$

      end for

      $$\nu_j(k) \longleftarrow \sum_{l=1}^{m(k)} \beta_{lj}\nu_{lj}(k)$$

      $\hat{x}_j(k|k) \longleftarrow \hat{x}_j(k|k-1) + K_j(k)\nu_j(k)$

      $P_j^c(k|k) \longleftarrow P_j(k|k-1) - K_j(k)S_j(k)K_j(k)'$

      $$\tilde{P}_j(k) \longleftarrow K_j(k)\Big[\sum_{l=0}^{m(k)} \beta_{lj}(k)\nu_{lj}(k)\nu_{lj}(k)' - \nu_j(k)\nu_j(k)'\Big]K_j(k)'$$

      $P_j(k|k) \longleftarrow \beta_{0,j}(k)P_j(k|k-1) + [1 - \beta_{0,j}(k)]P_j^c(k|k) + \tilde{P}_j(k)$

    Prediction

      $\hat{x}_j(k+1|k) \longleftarrow A(k)\hat{x}_j(k|k)$

      $P_j(k+1|k) \longleftarrow A(k)P_j(k|k)A'(k) + D(k)Q(k)D(k)'$

    end for

  end for

  return prediction and correction sequence of $[\hat{x}, P]$

end function

Chapter 7

Centralized Multitarget Tracking

In this chapter the multisensor problem is addressed. Specifically, two techniques for centralized update, described in [2, 27, 31], are discussed first, for a single target, under the assumption that the system is linear, and subsequently, the Parallel CJPDA filter and the Sequential CJPDAF are described for a multitarget scenario. The term *centralized* states that a central data processor receives all the measurements for filtering.

The multisensor problem is to track a target in clutter with N_S sensors. Assuming that the sensors are synchronized, at every sampling interval the measurements received by the N_S sensors are collected in a central processor.

The target dynamics and the measurements are assumed to obey the following linear equations:

$$\begin{cases} x(k+1) = A(k)x(k) + w(k) \\ \quad z^i(k) = C^i(k)x(k) + v^i(k) \qquad i = 1, \ldots, N_S \end{cases} \tag{7.1}$$

The measurement noise sequences are zero-mean, white, independent of the process noise $w(k)$ and independent from sensor to sensor. For this reason,

the covariances are

$$E\left[v^i(k)v^j(l)'\right] = R^i(k)\delta_{ij}\delta_{kl} \tag{7.2}$$

Notice that, for linear systems, parallel and sequential Kalman filtering of measurements from multiple sensors are equivalent and optimum [31]. When multiple sensors are used for tracking the states of multiple objects in a cluttered environment, a data association is necessary in order to assign measurements to the objects. When probabilistic data association algorithms are used, even in the case of linear systems, since association probabilities β_{jl} depend in a nonlinear way on measurements, the equivalence of parallel and sequential implementations no longer hold and it is not obvious which method would yield better tracking performance. Simulations results with both parallel and sequential implementation of a multisensor JPDA algorithm, suggest that the sequential method yields better tracking performance [27].

7.1 Parallel Centralized Fusion

In the parallel updating approach, all the measurements from all the sensors form a stacked vector and the state is updated simultaneously using this vector containing the whole set of validated measurements . This vector is

$$z(k) = \begin{bmatrix} z^1(k) \\ \vdots \\ z^{N_S}(k) \end{bmatrix} = C(k)x(k) + v(k) \tag{7.3}$$

where

$$C(k) = \begin{bmatrix} C^1(k) \\ \vdots \\ C^{N_S}(k) \end{bmatrix} \tag{7.4}$$

$$v(k) = \begin{bmatrix} v^1(k) \\ \vdots \\ v^{N_S}(k) \end{bmatrix} \tag{7.5}$$

In addition, if the measurement noise for different sensors are uncorrelated, the covariance matrix of the stacked noise vector $z(k)$ is

$$E\left[v(k)v(k)'\right] = R(k) = diag\left[R^i(k)\right] = \tag{7.6}$$

$$= \begin{bmatrix} R^1(k) & 0 & \dots & 0 \\ 0 & \ddots & 0 & \vdots \\ \vdots & 0 & \ddots & 0 \\ 0 & \dots & 0 & R^{N_S}(k) \end{bmatrix}$$

The state update equation is

$$\begin{aligned} \hat{x}(k|k) &= \hat{x}(k|k-1) + K(k)\nu(k) \\ &= \hat{x}(k|k-1) + P(k|k)C(k)'R(k)^{-1}\nu(k) \end{aligned} \tag{7.7}$$

where the following Kalman filter gain expression has been used

$$K(k) = P(k|k)C(k)'R(k)^{-1} \tag{7.8}$$

and the stacked innovation is

$$\begin{aligned} \nu(k) &= \begin{bmatrix} z^1(k) - \hat{z}^1(k|k-1) \\ \vdots \\ z^{N_S}(k) - \hat{z}^{N_S}(k|k-1) \end{bmatrix} \\ &= \begin{bmatrix} z^1(k) - C^1(k)\hat{x}(k|k-1) \\ \vdots \\ z^{N_S}(k) - C^{N_S}(k)\hat{x}(k|k-1) \end{bmatrix} \end{aligned} \tag{7.9}$$

where $\hat{z}^i(k|k-1)$ denotes the predicted measurement for sensor i. Using the block-diagonal form of the matrix $R(k)$, the updated state can be rewritten as

$$\begin{aligned} \hat{x}(k|k) &= \hat{x}(k|k-1) + \sum_{i=1}^{N_S} K^i(k)\nu^i(k) \\ &= \hat{x}(k|k-1) + P(k|k)\sum_{i=1}^{N_S} C^i(k)'R^i(k)\nu^i(k) \end{aligned} \tag{7.10}$$

with

$$\nu^i(k) \triangleq z^i(k) - C^i(k)\hat{x}(k|k-1) \tag{7.11}$$

Similarly, the inverse covariance is

$$P(k|k)^{-1} = P(k|k-1)^{-1} + \sum_{i=1}^{N_S} C^i(k)' \left[R^i(k)\right]^{-1} C^i(k) \tag{7.12}$$

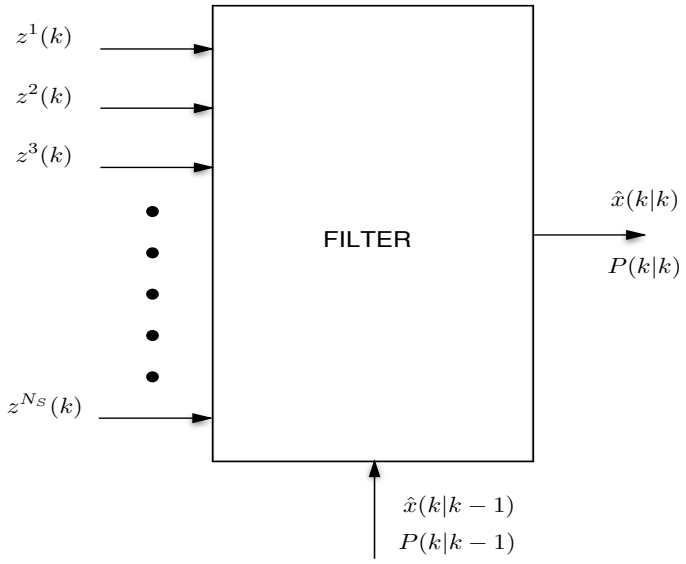

Figure 7.1: Parallel Filter.

7.2 Sequential Centralized Fusion

Under the assumption that measurements across time and sensors are uncorrelated, as expressed in (7.2), the update can be carried out sequentially with the measurement of one sensor at a time, so that the measurement of each sensor is used to further improve the intermediate state estimate.

The predicted state at time k and its covariance are

$$\begin{aligned} \hat{x}^0(k|k) &\triangleq \hat{x}(k|k-1) \\ P^0(k|k) &\triangleq P(k|k-1) \end{aligned} \tag{7.13}$$

Then the sequential state updates using the measurements $z^i(k)$ from (7.1), are

$$\hat{x}^i(k|k) = \hat{x}^{i-1}(k|k) + K^i(k)\nu^i(k) \qquad i = 1, \ldots, N_S \tag{7.14}$$

where

$$\nu^i(k) \triangleq z^i(k) - C^i(k)\hat{x}^{i-1}(k|k-1) \tag{7.15}$$

and the intermediate covariance updates are

$$P^i(k|k) = P^{i-1}(k|k-1) - K^i(k)S^i(k)K^i(k)' \qquad i = 1, \ldots, N_S \tag{7.16}$$

where

$$S^i(k) = C^i(k)P^{i-1}(k|k)C^i(k)' + R^i(k)K^i(k) = P^{i-1}(k|k)C^i(k)' \left[S^i(k)\right]^{-1} \tag{7.17}$$

The final updated estimate and covariance at time k are

$$\begin{aligned} \hat{x}(k|k) &= \hat{x}^{N_S}(k|k) \\ P(k|k) &= P^{N_S}(k|k) \end{aligned} \tag{7.18}$$

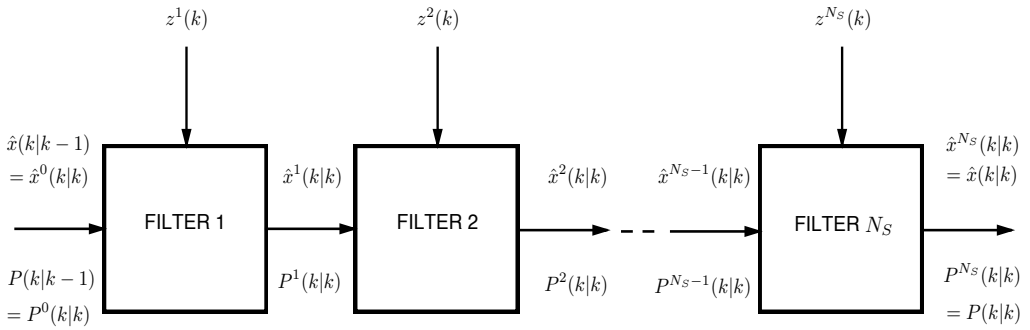

Figure 7.2: Sequential Filter.

7.3 Parallel CJPDAF

In the next two sections parallel and sequential implementations of the multisensor Cheap Joint Probabilistic Data Association Filter (CJPDAF) tracking algorithm are described.

The parallel CJPDAF uses the parallel multisensor update, mentioned in Section 7.1. Assuming a multitarget scenario, the update is preceded by the multisensor data association. The method chosen is CJPDA, presented in Section 6.4.3, which is carried out with all the measurements taken into account, so that the targets are associated with the measurements provided by each node.

The target dynamics and the measurements from sensor i are modeled as

$$\begin{cases} x_j(k+1) = f(k, x_j(k)) + w_j(k) & j = 1, \ldots, N_T \\ z_j^i(k) = h^i(k, x_j(k)) + v_j^i(k) & i = 1, \ldots, N_S \qquad j = 1, \ldots, N_T \end{cases} \tag{7.19}$$

where $x_j(k)$ denotes the state vector of target j at the kth scan, and $z_j^i(k)$ denotes the target-originated measurement from sensor i. The functions $f(k, x_j(k))$ and $h^i(k, x_j(k))$ and the noise covariances are assumed to be known. After a gating process, let $m^i(k)$ denote the number of validated measurements from sensor i at time k. Note that for a given target j and a sensor i, it is not known which measurement l ($1 \leq l \leq m^i(k)$) originates from the target. This is the reason why data association is necessary to associate the N_T targets with the $m^i(k)$ measurements for each of the N_S sensors.

The algorithm is described by the following steps:

- ***Initialization*** The tracks are initialized via M/N logic.
- ***Prediction*** The predicted state $\hat{x}_j(k|k-1)$ and its covariance $P_j(k|k-1)$ are computed using $\hat{x}_j(k-1|k-1)$ and $P_j(k-1|k-1)$ for each track j. In this algorithm the Unscented Kalman Filter described in Section 3.2 is applied for filtering. In addition, also the filter gain $K_j^i(k)$, the predicted measurements, $z_j^i(k|k-1)$, and the corresponding covariances $S_j^i(k)$ are calculated for each sensor.

- ***Measurement validation*** Once computed $z_j^i(k|k-1)$ and $S_j^i(k)$, the measurements can be validated for $i = 1, \ldots, N_S$. The number of validated measurements will be $m^i(k)$ for each sensor.

- ***CJPDA*** The validated measurements are associated with a certain probability to the targets, as illustrated in Section 6.4.3. In particular using equation (6.80), at current time k one has:

$$\beta_{lj}^i = \frac{G_{lj}^i}{\sum_{k=1}^{N_T} G_{lk}^i + \sum_{k=1}^{m^i} G_{kj}^i - G_{lj}^i + B^i} \tag{7.20}$$

  Using the association probabilities $\beta_{lj}^i(k)$, for each sensor the combined innovation is obtained as:

$$\nu_j^i(k) = \sum_{l=1}^{m^i(k)} \beta_{lj}^i(k)\nu_{lj}^i(k) \qquad j = 1, \ldots, N_T \qquad i = 1, \ldots, N_S \tag{7.21}$$

  The combined innovation is computed for each track as a weighted sum, with the association probabilities, computed by the CJPDA, as weights.

- ***Parallel update*** The updated states of each track are then calculated in parallel, using (7.10):

$$\hat{x}_j(k|k) = \hat{x}_j(k|k-1) + \sum_{i=1}^{N_S} K_j^i(k)\nu_j^i(k) \qquad j = 1, \ldots, N_T \tag{7.22}$$

  where $K_j^i(k)$ is the filter gain for measurements from the ith sensor and $\nu_j^i(k)$ is the combined innovation for track j, calculated with measurements from sensor i. The corresponding updated covariance is

$$P_j(k|k) = \beta_{0j}(k)P_j(k|k-1) + [1 - \beta_{0j}(k)]P_j^c(k|k) + \tilde{P}_j(k) \tag{7.23}$$

  where

$$P_j^c(k|k) = P_j(k|k-1) - \sum_{i=1}^{N_S} K_j^i(k)S_j^i(k)K_j^i(k)' \tag{7.24}$$

is the covariance of the state updated with the correct measurement given by (6.41), and

$$\tilde{P}_j(k) = \sum_{i=1}^{N_S} \tilde{P}_j^i(k) \tag{7.25}$$

where

$$\tilde{P}_j^i(k) = K_j^i(k) \left[\sum_{l=1}^{m^i(k)} \beta_{lj}^i(k) \nu_{lj}^i(k) \nu_{lj}^i(k)' - \nu_j^i(k) \nu_j^i(k)' \right] K_j^i(k)' \tag{7.26}$$

is the spread of the innovations corresponding to target j and sensor i, already calculated in (6.49). Note that this term will be zero only if one of the $m^i(k)$ association probabilities $\beta_{lj}^i(k)$ is unity, i.e. there is no uncertainty on the measurement origin.

7.4 Sequential CJPDAF

This multisensor multitarget tracker is implemented with the same steps described in the previous section, but using a different update approach, which is performed as in Section 7.2, so that at each time instant, every sensor corrects the state estimate sequentially. The order of updating is chosen randomly at each scan.

The sequential CJPDAF is described by the following processes:

- ***Initialization*** The tracks are initialized through a M/N logic, which after M detections out of N scans, confirms a track.

- ***Prediction*** At scan k the predicted state $\hat{x}_j(k|k-1)$and its covariance $P_j(k|k-1)$ are calculated for each track, using the UKF prediction step with $\hat{x}_j(k-1|k-1)$ and $P_j(k-1|k-1)$ known. Then, according to (7.13), one has

$$\begin{aligned} \hat{x}_j^0(k|k) &\triangleq \hat{x}_j(k|k-1) \\ P_j^0(k|k) &\triangleq P_j(k|k-1) \end{aligned} \tag{7.27}$$

Subsequently, the remaining steps are carried out sequentially for $i = 1, \ldots, N_S$. The predicted measurements $z_j^i(k|k-1)$ and the corresponding covariances $S_j^i(k)$ are calculated for each track j. Note that the predicted measurement is given by:

$$z_j^i(k|k-1) = h^i(k, \hat{x}_j^{i-1}(k|k)) \tag{7.28}$$

The above expression shows that the state estimate corrected by sensor $i-1$ is used as predicted state estimate by sensor i.

- ***Measurement validation*** Using $z_j^i(k|k-1)$ and $S_j^i(k)$, the measurements from sensor i are validated.

- ***CJPDA*** The data association is performed for sensor i and the combined innovations are calculated using the association probabilities.

- ***Sequential update*** The state update is performed sequentially by correcting the state estimate computed with the previous sensor:

$$\hat{x}_j^i(k|k) = \hat{x}_j^{i-1}(k|k) + K_j^i(k)\nu_j^i(k) \qquad i = 1, \ldots, N_S \tag{7.29}$$

The covariance update yields

$$P_j^i(k|k) = P_j^{i-1}(k|k) - K_j^i(k)S_j^i(k)K_j^i(k)' \qquad i = 1, \ldots, N_S \tag{7.30}$$

After the last sensor update at time k, the final estimate and covariance are obtained as:

$$\begin{aligned} \hat{x}_j(k|k) &= \hat{x}_j^{N_S}(k|k) \\ P_j(k|k) &= P_j^{N_S}(k|k) \end{aligned} \tag{7.31}$$

Chapter 8

Distributed Multitarget Tracking

8.1 Sensor Networks

Sensor networks have received significant attention in recent years because of their huge potential in applications, and the considerable technical challenges they present. The network model considered in the present work consists of heterogeneous and geographically dispersed tracking agents (nodes) which have processing, communication and sensing capabilities. Specifically, each node not only can acquire measurements of kinematic variables of targets crossing the surrounding area, but it can also process local information as well as exchange data with neighbors. The network under consideration has the following features:

- there is no central fusion node;
- nodes are unaware of the total number of agents and the connections between them.

The network can be described in terms of a direct graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ where $\mathcal{N}$ is the set of nodes and $\mathcal{A} \subseteq \mathcal{N} \times \mathcal{N}$ is the set of arcs, representing links.

In particular, (i, j) belongs to $\mathcal{A}$ if node j can receive data from node i. For each node $j \in \mathcal{N}$, $\mathcal{N}^j \triangleq \{i \in \mathcal{N} : (i, j) \in \mathcal{A}\}$ denotes its set of neighbors, i.e. the set of nodes from which node j can receive data. By definition, $(j, j) \in \mathcal{A}$, for any node $j \in \mathcal{N}$ and, hence, $j \in \mathcal{N}^j$, $\forall j$. The total number of nodes in the network will be denoted by $|\mathcal{N}|$, the cardinality of $\mathcal{N}$.

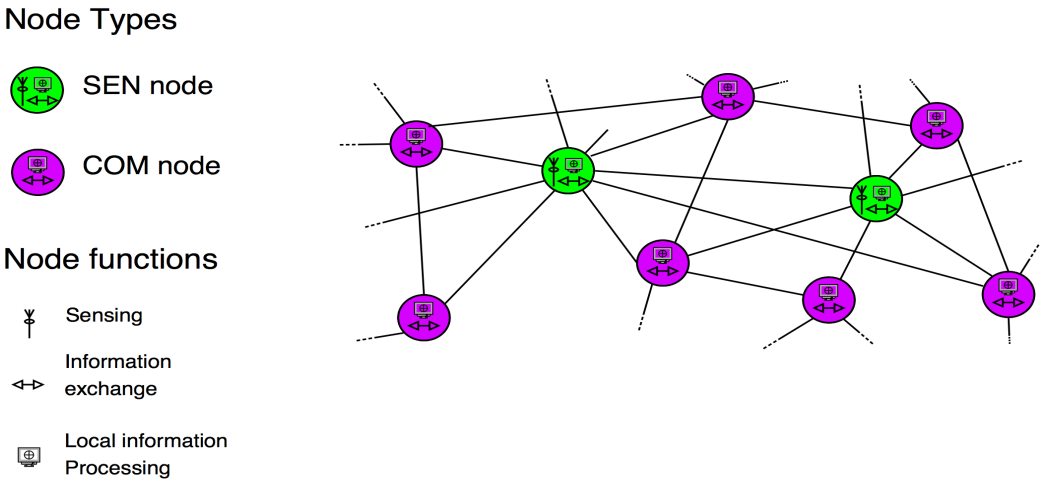

Figure 8.1: Network model.

Let us now describe how wireless links are established between nodes ([6],[7],[8]). Consider two nodes; the first transmits a signal with power p_t and the other receives this signal with power p_r. The signal is received properly if p_r is larger than or equal to a certain threshold power $p_{r,th}$, namely the *receiver sensitivity*. Thus, the sender establishes a wireless link to the receiver if $p_r \geq p_{r,th}$. The signal attenuation from the sender to the receiver, $\beta = \frac{p_t}{p_r}$, can be expressed, in decibel, as

$$\beta = 10 \log_{10} \left(\frac{p_t}{p_r} \right) \tag{8.1}$$

Given p_t and $\mathrm{p}_{r,th}$, two nodes can communicate via a direct link (i.e. they are neighbors) if the attenuation between them satisfies $\beta \leq \beta_{th}$, where

$$\beta_{th} = 10 \log_{10} \left(\frac{p_t}{p_{r,th}} \right) \tag{8.2}$$

is the threshold attenuation. The attenuation β arises as a consequence of two losses of the wireless channel:

- Path loss caused by distance;
- Shadow fading.

A simple model to describe the wireless channel assumes that the attenuation is only proportional to the distance d from the sending node, so that

$$p_r = \left(\frac{d}{C}\right)^{-\alpha} p_t \tag{8.3}$$

where C is a constant that represents the attenuation at a unit distance and α is the *path loss exponent* of the environment (e.g. $\alpha \simeq 2$ in free space and $\alpha \simeq 3$ in an urban outdoor environment). The attenuation in dB is, therefore, modeled as

$$\beta_0 = 10\,\alpha \log_{10}\left(\frac{d}{C}\right) \tag{8.4}$$

Using omnidirectional antennas, a node communicates to all nodes that are located within a circle of radius

$$d_{max} = \left(\frac{p_t}{p_{r,th}}\right)^{\frac{1}{\alpha}} \qquad [\text{m}] \tag{8.5}$$

around its position. This purely geometric model, in which two nodes are linked together if they are not further apart than a given threshold distance, the *transmission range* d_{max}, is shown in Figure 8.2.

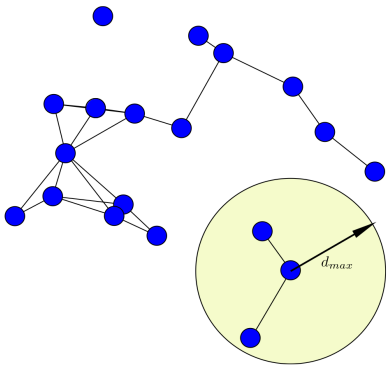

Figure 8.2: Geometric link model.

In environments with buildings, cars, walls, etc. there are other factors, apart from the distance, that contribute to the signal attenuation. This is due to the fact that several objects "shadow" the signal in different ways. Hence, receivers located at the same distance d from the sender may experience different values for p_r. The received power is then approximated by a probability density around a mean given by (8.3). Converting this probability density to dB, a Gaussian PDF is obtained as follows

$$f_{\beta_s}(\beta_s) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{\beta_s^2}{2\sigma^2}} \tag{8.6}$$

Typical values are up to 10 dB. Combining path loss and shadowing, the overall attenuation is given by

$$\beta = \beta_0 + \beta_s \tag{8.7}$$

where β_0 is a geometric, purely deterministic, component and β_s a purely random component. This link model is called shadow fading; an example is shown in Figure 8.3.

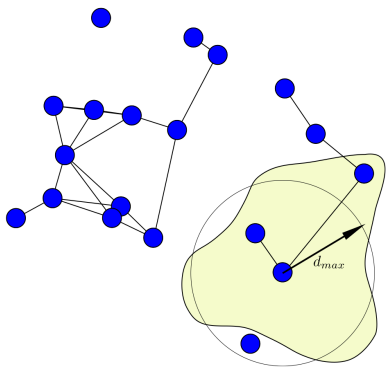

Figure 8.3: Shadow fading link model.

8.2 Track-to-Track Association

In distributed systems, once the state estimates and the corresponding covariances of the tracks in each node of the network are updated, a data

exchange between communicating nodes is carried out. As a consequence, an association process is needed to recognize tracks related to the same target, in order to perform the subsequent data fusion. As shown in [10], a typical track-to-track association technique first computes a distance metric between all the potential pairings, at the current time. Thereafter, the statistical distance is used in gating tests in order to select only the likely potential pairings and, finally, a track-to-track assignment matrix is formed to find the best matching solution.

8.2.1 Distance Metric

As mentioned before, the first step is to define a distance measure using directly Cartesian position and velocity estimates. Using the state estimates and the covariance matrices of two tracks (i, j), at the same time instant, the state difference vector is calculated as follows

$$\tilde{x}_{ij} = \hat{x}_i - \hat{x}_j \tag{8.8}$$

the above difference vector usually containing both position and velocity components. Thus, the statistical distance between track i and track j is defined as

$$d_{ij}^2 \triangleq \tilde{x}'_{ij} \left[P_i + P_j - P_{ij} - P'_{ij} \right]^{-1} \tilde{x}_{ij} = \tilde{x}'_{ij} S_{ij}^{-1} \tilde{x}_{ij} \tag{8.9}$$

where P_i and P_j are the state estimation covariance matrices for track i and track j respectively, whereas P_{ij} is the cross-covariance matrix between track i and track j estimation errors. Note that, in our case, the cross-covariance matrix is unknown. P_{ij} accounts for the correlation in the track errors caused by the common process noise, e.g. in a maneuvering target the two sensors can present similar lag errors.

8.2.2 The Track-to-Track Assignment Problem

In this section, the track-to-track association problem for two sensors is addressed, under the assumption that each node has only one single track per target. The followed approach is the same used for a general assignment matrix with supplementary conditions that are the existence of false tracks and missing tracks (one sensor is tracking a target that the other one has not yet detected). According to that, there is an additional row for sensor B assignment and an additional column for sensor A assignment, both accounting for the likelihood that there is no matching between a track from a sensor and tracks from the other sensor.

Consider d_{ij}^2, the normalized squared distance between track i from sensor A and track j from sensor B, defined in (8.9). The corresponding covariance matrix is denoted by S_{ij}. Then the following quantities are defined:

β_T = target density i.e. expected number of true targets divided by the volume of the field of view;

P_{TA}, P_{TB} = probability that sensors A, B have a track on a given target in the common field of view;

β_{FTA}, β_{FTB} = false track density for sensors A, B .

In addition, n is defined as the dimension used to calculate the statistical distance d_{ij}^2, e.g. if two Cartesian components of position and velocity are used, then $n = 4$.
The assignment matrix, shown in Table 8.1, is formed with the following elements:

$$P_{ij} = \frac{\beta_T P_{TA} P_{TB} e^{-d_{ij}^2/2}}{(2\pi)^{M/2}\sqrt{|S_{ij}|}} \tag{8.10}$$

$$P_{NTA} = \beta_T P_{TB}(1 - P_{TA}) + \beta_{FTB} \tag{8.11}$$

$$P_{NTB} = \beta_T P_{TA}(1 - P_{TB}) + \beta_{FTA} \tag{8.12}$$

P_{ij} represents the probability density function corresponding to targets with a track in both sensors, where d_{ij}^2 is the normalized distance between the tracks. P_{NTA} is the density of tracks from sensor B with no matching track from sensor A. Note that it is the sum of the density of true target tracks from sensor B with no corresponding track from sensor A, plus the density of false target track from sensor B. The interpretation is identical for P_{NTB}. If the assumed number of false tracks for both sensors are available, then dividing by the volume of the field of view, P_{NTA} and P_{NTB} are computed. The

Sensor A Tracks \ Sensor B Tracks	1	2	...	N_B	No Track
1	P_{11}	P_{12}	...	P_{1N_B}	P_{NTB}
2	P_{21}	P_{22}	...	P_{2N_B}	P_{NTB}
...	...	...	...	...	...
N_A	P_{N_A1}	P_{N_A2}	...	$P_{N_AN_B}$	P_{NTB}
No Track	P_{NTA}	P_{NTA}	...	P_{NTA}	

Table 8.1: General Track-to-Track Assignment Matrix

assignment matrix can be solved if the following operations are performed:

- Calculate the logarithm and multiply all terms by 2.
- Subtract the last row value $2\ln P_{NTA}$, no track element, from all terms
- Change the sign of all elements so that the problem turns into a cost minimization.
- Add $2\ln\left[\frac{\beta_T P_{TA} P_{TB}}{(2\pi)^{M/2} P_{NTA}}\right]$ to all non-zero elements left.
- Define the no assignment cost values:

$$C_N = 2\ln\left[\frac{\beta_T P_{TA} P_{TB}}{(2\pi)^{M/2} P_{NTA} P_{NTB}}\right] \tag{8.13}$$

Finally, the reduced assignment matrix is obtained, where the elements are the costs C_{ij} of assigning track i from sensor A to track j from sensor B:

$$C_{ij} = d_{ij}^2 + 2\ln\sqrt{|S_{ij}|} \tag{8.14}$$

Sensor A	Sensor B Tracks				No Track			
Tracks	1	2	...	N_B	1	2	...	N_A
1	C_{11}	C_{12}		C_{1N_B}	C_N	x	x	x
2	C_{21}	C_{22}		C_{2N_B}	x	C_N	x	x
$\vdots$	$\vdots$	$\vdots$		$\vdots$	x	x	$\ddots$	x
N_A	C_{N_A1}	C_{N_A2}		$C_{N_AN_B}$	x	x	x	C_N

Table 8.2: Cost Form of Track-to-Track Assignment Matrix

In Table 8.2, C_N is the cost of assigning no track of sensor B to a track of sensor A and x denotes an assignment that is not permitted. Furthermore, a threshold on the normalized distance between tracks can be chosen, so that if the threshold is exceeded, then the tracks are not associated, but rather the no assignment cost is chosen. So the assignment of track i from sensor A to track j from sensor B can occur only if $C_{ij} < C_N$:

$$d_{ij}^2 < 2\ln\left[\frac{\beta_T P_{TA} P_{TB}}{(2\pi)^{M/2} P_{NTA} P_{NTB}\sqrt{|S_{ij}|}}\right] \triangleq G_{ij} \tag{8.15}$$

G_{ij} is the gate for the assignment matrix, which, if the false track densities β_{FTA} and β_{FTB} are negligible, becomes

$$G_{ij} = 2\ln\left[\frac{1}{(2\pi)^{M/2}(1-P_{TA})(1-P_{TB})\beta_T\sqrt{|S_{ij}|}}\right] \tag{8.16}$$

since $P_{NTA} \cong \beta_T P_{TB}(1-P_{TA}), P_{NTB} \cong \beta_T P_{TA}(1-P_{TB})$ when $\beta_{FTA} \cong 0$ and $\beta_{FTB} \cong 0$. Note that the gate becomes very large as P_{TA} and/or P_{TB} approach unity; this accounts for the fact that the maximum number of

assignments can be done when at least a sensor has a track for each target and there are no false tracks in both nodes.

Once determined the cost form of the assignment matrix, track-to-track association is equivalent to the classical assignment problem, defined in Section 6.4.1, and can be solved with polynomial-time solutions, e.g. the Hungarian algorithm, described in Section 8.2.3.

8.2.3 Hungarian Algorithm

The assignment problem, also known as the maximum weighted bipartite matching problem, is a widely-studied problem applicable to many domains. It can be stated as follows: given a bipartite graph made up of two partitions V and U, and a set of weighted edges E between the two partitions, the problem requires the selection of a subset of the edges with a maximum sum of weights such that each node $v_i \in V$ or $u_i \in U$ is connected to at most one edge. The problem may also be phrased as a minimization problem by considering, instead of edge weights ω_{ij}, a set of non-negative edge costs, $c_{ij} = W\, w_{ij}$, where W is at least as large as the maximum of all the edge weights. Here we consider the minimization formulation of the problem.

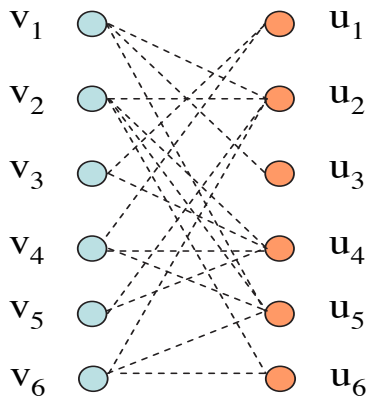

Figure 8.4: A bipartite graph.

The classical solution to the assignment problem is given by the Hungarian or Kuhn-Munkres algorithm, originally proposed by H. W. Kuhn in [22] and refined by J. Munkres in [26]. The Hungarian algorithm solves the assignment problem in $O(n^3)$ time, where n is the size of one partition of the bipartite graph. This and other existing algorithms for solving the assignment problem assume the a priori existence of a matrix of edge weights, w_{ij}, or costs, c_{ij}, and the problem is solved with respect to these values.

The Hungarian algorithm assumes the existence of a bipartite graph, $G = \{V, U, E\}$ where V and U are the sets of nodes in each partition of the graph, and E is the set of edges. Missing edges are assumed to have zero weight. The minimization form of the problem assumes a matrix of edge costs, $c_{ij} = W\, w_{ij}$ where $W \geq max(w_{ij})$. Missing edges may be given a large cost ($\geq W$). Each node in the graph may be matched (assigned) or unmatched (unassigned). Unmatched nodes are also called exposed.

The algorithm assigns dual variables α_i to each node v_i and dual variables β_j to each node v_j . It exploits the fact that the dual of the minimization version of the assignment problem is feasible when $\alpha_i + \beta_j \leq c_{ij}$ [28]. The Hungarian algorithm maintains feasible values for all the α_i and β_j from initialization through termination. An edge in the bipartite graph is called *admissible* when $\alpha_i + \beta_j = c_{ij}$. The subgraph consisting of only the currently admissible edges is called the *equality subgraph*. Starting with an empty matching, the basic strategy employed by the Hungarian algorithm is to repeatedly search for augmenting paths in the equality subgraph. If an augmenting path is found, the current set of matches is augmented by flipping the matched and unmatched edges along this path. Because there is one more unmatched than matched edge, this flipping increases the cardinality of the matching by one, completing a single stage of the algorithm. If an augmenting path is not found, the dual variables are adjusted to bring additional edges into the

equality subgraph by making them admissible, and the search continues. n such stages of the algorithm are performed to determine n matches, at which point the algorithm terminates.

If the size of the two partitions of the graph are not equal, a typical strategy is to insert into the relevant partition, dummy nodes with zero-weight edges to all nodes in the opposite partition. As such, the Hungarian algorithm always returns a complete matching, but this matching may include some zero-weight edges, representing "no assignment". Each stage of the Hungarian algorithm takes $O(n^2)$ arithmetic operations (if implemented with the appropriate data structures, and the computational complexity of the entire algorithm involving n stages is thus $O(n^3)$. The Hungarian algorithm is provably complete and optimal [28].

8.3 Consensus

In distributed algorithms, there is no central fusion center and the nodes of the sensor network do not have any global knowledge of the network topology. Consider the network model mentioned in Section 8.1 and suppose that, in each node of the network, after local processing, a PDF representing the local information is available. For example, such PDFs can be the result of some recursive Bayesian estimation algorithm (as in our case). The local information on the vector of interest is exchanged with neighbors in order to eventually reach an agreement on a good estimate. The objective is to implement a distributed fusion which guarantees scalability, i.e. the computational load in each node is independent of the size of the sensor network. Due to this requirement, Bayesian approaches are not suitable. As a matter of fact, they need to know, for each pair of neighbors, the quantity of interest conditioned to the common information. This is impossible to achieve in a scalable way. Hence, a robust suboptimal fusion technique, namely consensus, [5],[12] is used

for distributed averaging over a sensor network. The basic idea is that each node can compute the collective average of a given quantity by calculating iterative regional averages. The collective average takes into account all the nodes of the network, whereas the regional averages are evaluated between neighboring nodes only.

A consensus problem can be defined as follows. Consider node $i \in \mathcal{N}$ and the estimate $\hat{\theta}^i$ of a given quantity θ available at node i. The average consensus problem consists of finding an algorithm such that

$$\lim_{\ell \to \infty} \hat{\theta}^i(\ell) = \hat{\theta}, \quad \forall i \in \mathcal{N} \tag{8.17}$$

In other words, the objective of consensus algorithms is the convergence of regional averages to the following collective average

$$\bar{\theta} = \frac{1}{|\mathcal{N}|} \sum_{i \in \mathcal{N}} \hat{\theta}^i \tag{8.18}$$

starting from the initialization $\hat{\theta}^i(0) = \hat{\theta}^i$, then a simple consensus algorithm has the following iterative form

$$\hat{\theta}^i(\ell + 1) = \sum_{j \in \mathcal{N}^i} \pi^{i,j} \hat{\theta}^j(\ell) \quad \forall i \in \mathcal{N}, \tag{8.19}$$

that is the regional average computed in node i, where the *consensus weights* must fulfill the conditions

$$\begin{cases} \pi^{i,j} \geq 0 & \forall i,\, j \in \mathcal{N} \\ \displaystyle\sum_{j \in \mathcal{N}^i} \pi^{i,j} = 1 & \forall i \in \mathcal{N} \end{cases} \tag{8.20}$$

It can be noted from (8.19) and (8.70) that for each node the estimate at a given consensus step $\ell + 1$ is computed as a convex combination of the estimates of the neighbors at the previous consensus step ℓ.

8.3.1 Information Filter

In distributed state estimation algorithms, usually it is convenient to use the information filter (IF) instead of the standard Kalman filter, discussed in Chapter 1. The conventional (covariance) Kalman filter propagates in time the state estimate $\hat{x}(k|k-1)$ and the corresponding covariance $P(k|k-1)$. While the information filter updates the inverse of the covariance matrix, called information matrix, defined as:

$$\begin{aligned} \Omega(k|k-1) &\triangleq P(k|k-1)^{-1} \\ \Omega(k|k) &\triangleq P(k|k)^{-1} \end{aligned} \tag{8.21}$$

and the so called information vectors defined as follows:

$$\begin{aligned} q(k|k-1) &\triangleq P(k|k-1)^{-1}\hat{x}(k|k-1) \\ q(k|k) &\triangleq P(k|k)^{-1}\hat{x}(k|k) \end{aligned} \tag{8.22}$$

Under the assumption that the covariance matrix of the process noise Q is nonsingular, i.e. $\det(Q) \neq 0$, recursive equations for the information state vector and the information matrix, can be derived directly from the Kalman filter equations. The resulting information filter is mathematically equivalent to the standard Kalman filter. The correction equation for the information matrix is

$$\begin{aligned} \Omega(k|k) &= \Omega(k|k-1) + C'R^{-1}C \\ &= \Omega(k|k-1) + \sum_{i\in\mathcal{N}} (C^i)'(R^i)^{-1}C^i \end{aligned} \tag{8.23}$$

where $\mathcal{N}$ is the set of sensors, i.e.nodes that get measurements of the state of the system. A similar expression can be found for the information state vector

$$\begin{aligned} q(k|k) &= q(k|k-1) + C'R^{-1}z(k) \\ &= q(k|k-1) + \sum_{i\in\mathcal{N}} (C^i)'(R^i)^{-1}z^i(k) \end{aligned} \tag{8.24}$$

initialized by $\Omega(0|-1) = P(0|-1)^{-1}$ and $q(0|-1) = \Omega(0|-1)\hat{x}(0|-1)$, respectively. Note that, due to the independence of measurements, the filter update has the following form

$$\begin{aligned} \Omega(k|k) &= \Omega(k|k-1) + \sum_{i \in \mathcal{N}} \delta\Omega^i \\ q(k|k) &= q(k|k-1) + \sum_{i \in \mathcal{N}} \delta q^i(k) \end{aligned}$$

where

$$\begin{aligned} \delta\Omega^i &\triangleq \left(C^i\right)' \left(R^i\right)^{-1} C^i \\ \delta q^i(k) &\triangleq \left(C^i\right)' \left(R^i\right)^{-1} z^i(k) \end{aligned} \tag{8.25}$$

represent the additive corrections for the information matrix and the information vector provided by node i.

The information matrix prediction can be found by applying the matrix inversion lemma

$$(A + BCD)^{-1} = A^{-1} - A^{-1}B(C^{-1} + DA^{-1}B)^{-1}DA^{-1} \tag{8.26}$$

to the Kalman filter covariance prediction in (2.12), which yields

$$\Omega(k+1|k) = Q^{-1} - Q^{-1}A\left(\Omega(k|k) + A'Q^{-1}A\right)^{-1} A'Q^{-1} \tag{8.27}$$

Similarly, the information state vector prediction can be obtained as

$$q(k+1|k) = Q^{-1}A\left(\Omega(k|k) + A'Q^{-1}A\right)^{-1} q(k|k) \tag{8.28}$$

Algorithm 4 summarizes the information filter recursion.

Algorithm 6 IF

function IF($\hat{x}(0|-1)$, $P(0|-1)$, $\delta\Omega^i, \delta q^i(k)$) $\qquad i = 1, \dots, N_s$

  for all time $k = 0, 1, 2, \dots$ **do**

    Correction

    $\delta\Omega \longleftarrow C' R^{-1} C$

    $\Omega(k|k) \longleftarrow \Omega(k|k-1) + \sum_{i \in N_S} \delta\Omega^i$

    $q(k|k) \longleftarrow q(k|k-1) + \sum_{i \in N_S} \delta q^i(k)$

    $P(k|k) \longleftarrow \Omega(k|k)^{-1}$

    $\hat{x}(k|k) \longleftarrow P(k|k)\, q(k|k)$

    Prediction

    $\Omega(k+1|k) \longleftarrow Q^{-1} - Q^{-1} A \left(\Omega(k|k) + A' Q^{-1} A\right)^{-1} A' Q^{-1}$

    $q(k+1|k) \longleftarrow Q^{-1} A \left(\Omega(k|k) + A' Q^{-1} A\right)^{-1} q(k|k)$

    $P(k+1|k) \longleftarrow \Omega(k+1|k)^{-1}$

    $\hat{x}(k+1|k) \longleftarrow P(k+1|k) q(k+1|k)$

  end for

  return prediction and correction sequence of $[\hat{x}, P]$

end function

8.3.2 Consensus on Information

Recalling the definition of information matrix and information vector, the consensus problem (8.17) can be solved by updating the local data via convex combination with the data received by the neighbors. This choice leads to the following algorithm for consensus on information

$$\Omega^i(\ell+1) = \sum_{j \in \mathcal{N}^i} \pi^{i,j}\, \Omega^j(\ell) \qquad \forall i \in \mathcal{N}, \ell = 0, 1, \dots \tag{8.29}$$

$$q^i(\ell+1) = \sum_{j \in \mathcal{N}^i} \pi^{i,j}\, q^j(\ell) \qquad \forall i \in \mathcal{N}, \ell = 0, 1, \dots \tag{8.30}$$

where the recursion is initialized by $\Omega^i(0) = \Omega^i$ and $q^i(0) = q^i$, $\forall i \in \mathcal{N}$.
Let Π denote the consensus matrix whose (i, j)-element is the consensus weight $\pi^{i,j}$ (if $j \notin \mathcal{N}^i$ then $\pi^{i,j}$ is taken as 0). The non-negative square consensus matrix Π is primitive if there exists an integer m such that all the elements of Π^m are strictly positive. Furthermore Π is doubly stochastic if all its rows and columns sum up to unity, i.e.

$$\sum_{i \in \mathcal{N}} \pi^{i,j} = 1 \quad \forall j \in \mathcal{N} \tag{8.31}$$

$$\sum_{j \in \mathcal{N}} \pi^{i,j} = 1 \quad \forall i \in \mathcal{N} \tag{8.32}$$

Let the consensus matrix Π be primitive and doubly stochastic. Then, the consensus algorithm (8.29)-(8.30) leads asymptotically to the following convergence

$$\lim_{\ell \to \infty} \Omega^i(\ell) = \bar{\Omega}, \quad \forall i \in \mathcal{N} \tag{8.33}$$

$$\lim_{\ell \to \infty} q^i(\ell) = \bar{q}, \quad \forall i \in \mathcal{N} \tag{8.34}$$

where $\bar{\Omega}$ and $\bar{q}$ are the collective averages of the information matrices and, respectively, vectors. As discussed in [11], a necessary condition for the matrix Π to be primitive is that the graph $\mathcal{G}$ associated with the sensor network is strongly connected. Under this assumption, a possible choice for the consensus weights (8.70) is given by the *Metropolis weights* [23, 11]:

$$\pi^{i,j} = \frac{1}{\max\{|N_i|, |N_j|\}} \qquad \forall i, j \in \mathcal{N}, i \neq j$$

$$\pi^{i,i} = 1 - \sum_{j \in \mathcal{N}^i, j \neq i} \pi^{i,j} \qquad \forall i \in \mathcal{N}$$

The consensus on information is then applied to distributed state estimation. A linear-Gaussian case is first considered, which can later be extended to the

nonlinear case. Consider a discrete-time linear system

$$x(k+1) = Ax(k) + w(k) \tag{8.35}$$

whose state is measured by $\mathcal{N}$ linear sensors

$$z^i(k) = C^i x(k) + v(k) \qquad i \in \mathcal{N} \tag{8.36}$$

Let the initial state, the process disturbance and all the measurement noises be normally distributed. It is convenient to use the information filter recursion, dicussed in Section 8.3.1. Assuming that at time k, in each node $i \in \mathcal{N}$, the local information pair $(\Omega^i(k|k-1), q^i(k|k-1))$ is available, the correction can be written as

$$\begin{cases} \Omega^i(k|k) &= \Omega^i(k|k-1) + \left(C^i\right)' \left(R^i\right)^{-1} C^i \\ \\ q^i(k|k) &= q^i(k|k-1) + \left(C^i\right)' \left(R^i\right)^{-1} z^i(k) \end{cases} \tag{8.37}$$

and the prediction is

$$\begin{cases} \Omega^i(k+1|k) &= Q^{-1} - Q^{-1}A\left[\Omega^i(k|k,L) + A'Q^{-1}A\right]^{-1} A'Q^{-1} \\ \\ q^i(k+1|k) &= Q^{-1}A\left[\Omega^i(k|k,L) + A'Q^{-1}A\right]^{-1} q^i(k|k,L) \end{cases} \tag{8.38}$$

Note that the matrix Q has to be invertible so that equations in (8.38) make sense. Using the above information filter as well as consensus (8.29)-(8.30), the algorithm for distributed state estimation (Algorithm 7) at each time $k = 0, 1, \ldots$, and for each node $i \in \mathcal{N}$, performs the following steps

1. The local set of measurements $z^i(k)$ is collected and the local information pair $(\Omega^i(k|k-1), q^i(k|k-1))$ is updated using equation (8.37), in order to obtain the local posterior information pair $(\Omega^i(k|k,0), q^i(k|k,0))$.

2. For $\ell = 0, 1, \ldots, L-1$ the following consensus steps are performed

- Information $(\Omega^i(k|k,\ell), q^i(k|k,\ell))$ is exchanged between neighbors. Node i transmits its data to nodes j such that $i \in \mathcal{N}^j$.
- Node i waits for data to be received from each node $j \in \mathcal{N}^i$.
- Next, the fusion is carried out in node i as follows

$$\begin{aligned} \Omega^i(k|k,\ell+1) &= \sum_{j\in\mathcal{N}^i} \pi^{i,j}\, \Omega^j(k|k,\ell) \qquad \ell = 0,\ldots,L-1 \\ q^i(k|k,\ell+1) &= \sum_{j\in\mathcal{N}^i} \pi^{i,j}\, q^j(k|k,\ell) \qquad \ell = 0,\ldots,L-1 \end{aligned} \tag{8.39}$$

  to obtain the fused information pair

$$(\Omega^i(k|k), q^i(k|k)) \triangleq (\Omega^i(k|k,L), q^i(k|k,L)). \tag{8.40}$$

3. Finally, the local prior information pair $(\Omega^i(k+1|k), q^i(k+1|k))$ is calculated from $(\Omega^i(k|k,L), q^i(k|k,L))$ via the prediction equations (8.38).

Algorithm 7 CONSENSUS-IF

1: **function** CONSENSUS-IF($\Omega^i(0|-1)$, $q^i(0|-1)$)

2: **for all** time $k = 0, 1, 2, \ldots$ **do**

Correction

3: $\Omega^i(k|k) \longleftarrow \Omega^i(k|k-1) + (C^i)' (R^i)^{-1} C^i$

4: $q^i(k|k) \longleftarrow q^i(k|k-1) + (C^i)' (R^i)^{-1} z^i(k)$

5: $\Omega^i(k|k,0) \longleftarrow \Omega^i(k|k)$

6: $q^i(k|k,0) \longleftarrow q^i(k|k)$

7: $\left[\Omega^i(k|k,L),\, q^i(k|k,L)\right] \longleftarrow$ ***consensusOnInformation***

8: $\hat{x}^i(k|k) \longleftarrow [\Omega^i(k|k,L)]^{-1} q^i(k|k,L)$

Prediction

9: $\Omega^i(k+1|k) \longleftarrow Q^{-1} - Q^{-1}A\left[\Omega^i(k|k,L) + A'Q^{-1}A\right]^{-1} A'Q^{-1}$

10: $q^i(k+1|k) \longleftarrow Q^{-1}A\left[\Omega^i(k|k,L) + A'Q^{-1}A\right]^{-1} q^i(k|k,L)$

11: $\hat{x}^i(k+1|k) \longleftarrow [\Omega^i(k+1|k)]^{-1} q^i(k+1|k)$

12: **end for**

13: **return** prediction and correction sequence of $[\hat{x}, P]$

14: **end function**

Algorithm 8 consensusOnInformation

function CONSENSUSONINFORMATION($\Omega^i(k|k,0)$, $q^i(k|k,0)$)
  for all consensus steps $\ell = 0, 1, 2, \ldots, L-1$ **do**
    transmit$\Big(q^i(k|k,\ell),\, \Omega^i(k|k,\ell)\Big)$
    wait until $\forall j \in \mathcal{N}^i$, $\Big[q^j(k|k,\ell),\, \Omega^j(k|k,\ell)\Big]$ ***is received***
    for all $i, j \in \mathcal{N}$ $i \neq j$ **do**
      $\pi^{i,j} \longleftarrow \dfrac{1}{\max(N_i, N_j)}$
    end for
    for all $i \in \mathcal{N}$ **do**
      $\pi^{i,i} \longleftarrow 1 - \displaystyle\sum_{j \in \mathcal{N}^i\ j \neq i} \pi^{i,j}$
    end for
    $\Omega^i(k|k,\ell+1) \longleftarrow \displaystyle\sum_{j \in \mathcal{N}^i} \pi^{i,j}\, \Omega^j(k|k,\ell)$
    $q^i(k|k,\ell+1) \longleftarrow \displaystyle\sum_{j \in \mathcal{N}^i} \pi^{i,j}\, q^j(k|k,\ell)$
  end for
  return $\Big[\Omega^i(k|k,L),\, q^i(k|k,L)\Big]$
end function

Note that the fusion of the information received from multiple sensors (8.39), through a convex combination of the information matrices and vectors, coincides with the well-known Covariance Intersection fusion [19]. Therefore, this fusion rule corresponds to a single step of the consensus algorithm which computes, in a distributed fashion, the Kullback-Leibler average (KLA) of the local posterior PDFs (the KLA of N_S Gaussian PDFs can be simply obtained by averaging their information matrices and information vectors). This is discussed in [12].

In a distributed nonlinear state estimation, both correction and prediction steps are performed using a nonlinear filter, such as UKF, instead of the standard Kalman filter.

So far, the single-target information fusion problem via consensus has been addressed. The above discussion can be extended to the multitarget case through random set theory, as described in [5]. Let $f(\mathcal{X})$ denote the multiobject (or multitarget) density, which is the multitarget counterpart of the state PDF in single-target tracking, such that

$$\int_{\mathbb{X}} f(\mathcal{X})\delta\mathcal{X} = 1 \tag{8.41}$$

where $\mathbb{X}$ is the state space. Since the multiobject density involves a combinatorial complexity, simpler, even if incomplete, characterizations are usually adopted. For this reason, the first-order moment of the multiobject density, called Probability Hypothesis Density (PHD), can be used instead. The PHD function of $\mathcal{X}$ over $\mathbb{X}$ can be defined as

$$d(x) = \bar{n}s(x) \tag{8.42}$$

where $\bar{n}$ is the expected number of targets and $s(\cdot)$ is a single-target PDF, called location density.

Suppose that a multiobject density $f^i(\mathcal{X})$, computed on the basis of the information collected locally or propagated from other nodes, is available in each node i of the sensor network. Then, in order to combine the information from all neighboring nodes, a fusion step is carried out. Recalling the consensus problem (8.17), the average of the local multiobject densities $f^i(\mathcal{X})$ can be defined as the weighted Kullback-Leibler average $f_{KLA}(\mathcal{X})$, which turns out to be given by [5]

$$f_{KLA}(\mathcal{X}) = \frac{\prod_i \left[f^i(\mathcal{X})\right]^{\omega_i}}{\int \prod_i \left[f^i(\mathcal{X})\right]^{\omega_i} \delta\mathcal{X}} \tag{8.43}$$

which corresponds to the normalized weighted geometric mean of the node multiobject densities $f^i(\mathcal{X})$. Note that ω_i are weights satisfying

$$\omega_i \geq 0, \qquad \sum_i \omega_i = 1. \tag{8.44}$$

It is important to point out that the above fusion rule coincides with the Generalized Covariance Intersection (GCI) for multiobject fusion, first presented by Mahler [24]. The term Generalized Covariance Intersection originates from the fact that (8.43) is the multitarget counterpart of the fusion rule for single-target PDFs (8.39), which is also a generalization of the Covariance Intersection (CI), first devised for Gaussian PDFs [19]. Consider the estimates $\hat{x}^i$, of the same quantity x, from multiple estimators and its corresponding covariance P^i, their CI fusion is given by

$$\begin{aligned} P &= \left[\sum_i \omega^i (P^i)^{-1}\right]^{-1} \\ \hat{x} &= P \sum_i \omega^i (P^i)^{-1} \hat{x}^i \end{aligned} \tag{8.45}$$

The above fusion is equivalent to

$$p(x) = \frac{\prod_i \left[p^i(x)\right]^{\omega_i}}{\int \prod_i \left[p^i(x)\right]^{\omega_i} dx} \tag{8.46}$$

under the assumption of normally distributed estimates. Moreover, the above equivalence is valid for any choice of the weights ω_i satisfying (8.44) and only if all estimates are consistent, so that, given

$$E\left[(x - \hat{x}^i)(x - \hat{x}^i)'\right] \leq P^i \qquad \forall i \tag{8.47}$$

then, also the fused estimates are consistent

$$E\left[(x - \hat{x})(x - \hat{x})'\right] \leq P \tag{8.48}$$

In (8.46) $p(\cdot) \triangleq (\cdot; \hat{x}^i, P^i)$ is the Gaussian PDF with mean $\hat{x}^i$ and covariance P^i.

8.3.3 Gaussian Mixture Implementation

In the present work, the distributed multiobject fusion of the information is performed adopting the Gaussian Mixture (GM) approach. This choice

allows the avoidance of random set theory. Using the GM approach, the location PDFs in (8.42) can be expressed as linear combinations of Gaussian components, as follows

$$s(x) = \sum_{j=1}^{N_G} \alpha_j \mathcal{N}(x; \hat{x}_j, P_j) \tag{8.49}$$

Consider two nodes a and b, which provide the two GM location densities

$$s^i(x) = \sum_{j=1}^{N_G^i} \alpha_j^i \mathcal{N}(x; \hat{x}_j^i, P_j^i) \qquad i = a, b \tag{8.50}$$

Then, the fused location PDF is obtained as

$$\bar{s}(x) = \frac{[s^a(x)]^\omega \left[s^b(x)\right]^{1-\omega}}{\int [s^a(x)]^\omega \left[s^b(x)\right]^{1-\omega} dx} \tag{8.51}$$

Note that the above fusion, which involves exponentiation and multiplication of GMs, is not a Gaussian mixture. This can be demonstrated by the following observations about basic operations with Gaussian components and mixtures:

- The power of a Gaussian component is a Gaussian component

$$[\alpha \mathcal{N}(x; \hat{x}, P)]^\omega = \alpha^\omega \kappa(\omega, P) \mathcal{N}\left(x; \hat{x}, \frac{P}{\omega}\right) \tag{8.52}$$

  where

$$\kappa(\omega, P) \triangleq \frac{[det(2\pi P \omega^{-1})]^{\frac{1}{2}}}{[det(2\pi P)]^{\frac{\omega}{2}}} \tag{8.53}$$

- The product of Gaussian components is a Gaussian component, specifically for two components

$$\alpha_1 \mathcal{N}(x; \hat{x}_1, P_1) \cdot \alpha_2 \mathcal{N}(x; \hat{x}_2, P_2) = \alpha_{12} \mathcal{N}(x; \hat{x}_{12}, P_{12}) \tag{8.54}$$

  where

$$\begin{aligned} P_{12} &= (P_1^{-1} + P_2^{-1})^{-1} \\ \hat{x}_{12} &= P_{12}(P_1^{-1}\hat{x}_1 + P_2^{-1}\hat{x}_2) \\ \alpha_{12} &= \alpha_1 \alpha_2 \mathcal{N}(\hat{x}_1 - \hat{x}_2; 0, P_1 + P_2) \end{aligned} \tag{8.55}$$

- From (8.54) and the distributive property, it follows that the product of GMs is a GM. In particular, if $s^a(\cdot)$ and $s^b(\cdot)$ have N_G^a and N_G^b components respectively, then the product $s^a(\cdot)s^b(\cdot)$ will have $N_G^a N_G^b$ components.

- The exponentiation of a GM does not, in general, return a GM.

As a result of the last observation, the fusion in (8.51) is not a GM. In order to preserve a GM expression of the location PDF, even after the fusion, the following approximation can be used, instead of the standard GM exponentiation

$$\begin{aligned}\left[\sum_{j=1}^{N_G}\alpha_j\mathcal{N}(x;x_j,P_j)\right]^{\omega} &\cong \sum_{j=1}^{N_G}\left[\alpha_j\mathcal{N}(x;x_j,P_j)\right]^{\omega} \\ &= \sum_{j=1}^{N_G}\alpha_j^{\omega}\,\kappa(\omega,P_j)\,\mathcal{N}\left(x;x_j,\frac{P_j}{\omega}\right)\end{aligned} \tag{8.56}$$

Note that the above approximation is reasonable if the cross-products of the different terms in the GM are negligible for all x. As a matter of fact, the error in the approximation (8.56) decreases with the increase of the distance between the confidence ellipsoids of the Gaussian components. The conditions for the validity of (8.56) can be summarized using the Mahalanobis distance in the following inequalities

$$\begin{cases}(\hat{x}_i-\hat{x}_j)'P_i^{-1}(\hat{x}_i-\hat{x}_j)\gg 1\\ (\hat{x}_i-\hat{x}_j)'P_j^{-1}(\hat{x}_i-\hat{x}_j)\gg 1\end{cases} \tag{8.57}$$

Therefore, the above expression can be used to approximate the fusion (8.51) as follows

$$\bar{s}(x)=\frac{\displaystyle\sum_{i=1}^{N_G^a}\sum_{j=1}^{N_G^b}\alpha_{ij}^{ab}\mathcal{N}(x;\hat{x}_{ij}^{ab},P_{ij}^{ab})}{\displaystyle\int\sum_{i=1}^{N_G^a}\sum_{j=1}^{N_G^b}\alpha_{ij}^{ab}\mathcal{N}(x;\hat{x}_{ij}^{ab},P_{ij}^{ab})dx} \tag{8.58}$$

where

$$P_{ij}^{ab} = \left[\omega (P_i^a)^{-1} + (1-\omega)(P_j^b)^{-1}\right]^{-1} \tag{8.59}$$

$$\hat{x}_{ij}^{ab} = P_{ij}^{ab}\left[\omega (P_i^a)^{-1}\hat{x}_i^a + (1-\omega)(P_j^b)^{-1}\hat{x}_i^b\right]^{-1} \tag{8.60}$$

$$\begin{aligned}\alpha_{ij}^{ab} = (\alpha_i^a)^{\omega}(\alpha_j^b)^{1-\omega}\kappa(\omega, P_i^a)\kappa(1-\omega, P_j^b)\cdot \\ \cdot \mathcal{N}\left(\hat{x}_i^a - \hat{x}_j^b; 0, \frac{P_i^a}{\omega} + \frac{P_j^b}{1-\omega}\right)\end{aligned} \tag{8.61}$$

Equations (8.58)-(8.61) perform CI fusion on any pair formed by a Gaussian component of node a and a Gaussian component of node b. Note that the coefficient α_{ij}^{ab} includes the factor $\mathcal{N}\left(\hat{x}_i^a - \hat{x}_j^b; 0, P_i^a + P_j^b\right)$ which represents the distance between the two fusing components $(\hat{x}_i^a, P_i^a)$ and $(\hat{x}_j^b, P_j^b)$. Moreover, it is important to point out that Gaussian components with small coefficients α_{ij}^{ab} in (8.58), should be neglected. Specifically, fusing components whose Mahalanobis distance falls below a given threshold, i.e.

$$\sqrt{(x_i^a - x_i^b)'\left(\frac{P^a}{1-\omega} + \frac{P^b}{\omega}\right)^{-1}(x_i^a - x_i^b)} \leq \gamma_f \tag{8.62}$$

can be removed. Clearly, in order to fuse information provided by more than two sensors, the fusion (8.59) - (8.60) can be sequentially applied to every possible pair of nodes, for a total of $\mathcal{N} - 1$ times. The irrelevance of the chronological order used in the fusion is guaranteed by both associative and commutative properties of multiplication.

8.4 Consensus CJPDAF

In this section a solution to distributed multitarget tracking over a sensor network is presented. The architecture of the filter has been developed taking into account the following issues, already discussed throughout this thesis.

- Several measurements are detected for each track due to the presence of clutter and each measurement might be validated for multiple tracks.

Thus, a reliable and appropriate technique for multitarget data association in clutter is required.

- Each node has limited computational and sensing (e.g. TOA, DOA sensors) capabilities. Moreover, data transmission has to be limited because it is the main responsible for energy consumption.
- Information has to be processed in a distributed fashion, without any central unit and in a scalable way with respect to the network size.
- The single node is unaware of the network connections.
- The network can be formed by nonlinear sensors, hence a filter for nonlinear estimation is required.

The idea has been to design a tracker which combines different techniques so that each sub-problem can be solved. Taking into account the above considerations, first, the Joint Probabilistic Data Association has been implemented for measurement-to-track association, as it provides good performance in presence of high clutter and noisy models. Hard decision approaches, such as Global Nearest Neighbor, lack of robustness, and turn out to be more appropriate in low-clutter environments. In addition, in order to reduce the computational burden, the cheap version of JPDA (CJPDA) has been chosen. The only difference with JPDA is that instead of exactly calculating the association probabilities, the CJPDA approximates them. Nevertheless, it provides a satisfactory association performance. Due to the fact that this association method assumes that the track has been already initialized, also a track initiator has been implemented. Tracks are created by two-point differencing and subsequently confirmed through an M/N logic. The scalability requirement excludes the centralized fusion from the possible solutions and calls for a consensus approach in order to achieve global fusion over the whole

network by iterating local fusion steps among neighboring nodes. Furthermore, the impossibility to single out common information between nodes, requires robust fusion rules, such as Covariance Intersection (CI). Lastly, due to the nonlinearity of the employed sensors, the Unscented Kalman Filter is exploited in each sensor in order to update means and covariances.
Hence, the proposed Consensus CJPDAF approach exploits consensus in order to provide a distributed, scalable solution and to guarantee computational efficiency. In particular, two different variants of this approach are proposed. The first one, simply named C-CJPDAF (Consensus-CJPDA Filter), carries out a track-to-track association, via the solution of an assignment problem, before fusion. Due to limited processing-communication capabilities of the sensor network, parsimonious representations of the multitarget information transmitted over the radio links, are required. For this reason, the other proposed algorithm, named CGM-CJPDAF, relies on a Gaussian Mixture (GM) representation of the multitarget information in order to reduce computational and communication costs.

8.4.1 Consensus CJPDAF with Track-to-Track Association (C-CJPDAF)

In this distributed multitarget tracking algorithm all the aforementioned techniques are implemented. A brief description of the sequence of steps included in the C-CJPDAF is in order:

1. ***Local UKF prediction.*** First, each node i performs a local prediction of the state estimates and covariances corresponding to each confirmed and preliminary track, exploiting the single-target dynamics and the measurement equations.

2. ***Local measurement validation.*** The local measurement set Z^i is associated to existing tracks via a gating procedure based on the nor-

malized squared innovation (6.12) and following the priorities discussed in Section 6.1.2. In particular, there are two validation steps:

(a) ***Validation for confirmed tracks.*** Measurements Z^i are associated to confirmed tracks. Tracks may be deleted after a given number of consecutive misdetections.

(b) ***Validation for preliminary tracks.*** Measurement validation for the subset Z^i_{NC} is then performed. A preliminary track, which validates at least one measurement, can either be promoted to confirmed track, or remain preliminary. The decision is taken by the M/N logic. If no measurement is validated, the track can be kept or deleted.

3. ***Local TPD gating.*** Tentative new tracks, created at the previous scan, are compared to the subset Z^i_{NP} of unassociated measurements, in order to find matchings for the initialization of new tracks. New preliminary tracks are not updated and they are directly deferred to the next recursion.

4. ***Local CJPDA.*** The measurement-to-track association, which calculates the association probabilities for each track, is carried out following the cheap joint probabilistic data association, described in Section 6.4.3. The data association is performed for both confirmed and preliminary tracks, whose validated measurements are collected in the same vector.

5. ***Local UKF correction.*** State estimates and covariances are corrected through the unscented Kalman filter, exploiting the validated measurements and the corresponding association probabilities, given, respectively, by step 2 and 4. Once corrected the state estimates, tracks exceeding a reasonable velocity limitation are terminated.

6. ***Consensus.*** Each node i involves the subnetwork $\mathcal{N}^i$ in order to perform consensus. Specifically, the following operations take place:

   (a) ***Information exchange.*** Each node exchanges information with neighboring nodes. Agent i transmits its data to node $j \in \mathcal{N}^j$ and receives information from $j \in \mathcal{N}^i$.

   (b) ***Track-to-track association.*** Once the information has been exchanged between neighbors, each node carries out a pairwise track-to-track association procedure, which is essentially an assignment problem, as discussed in Section 8.2.2, that can exactly be solved in polynomial time by the Hungarian method.

   (c) ***CI fusion.*** When data originating from the same track have been associated, the information is fused using equations (8.58)-(8.61).

After the fusion, confirmed tracks after consensus are displayed on screen, and then they will start the local recursion for the following scan, together with unassociated tracks in step 6(b), which are kept locally.
The C-CJPDAF steps are summarized in Algorithm 9. Note that, due to the nonlinearity of the sensors used in Chapter 9 for simulations, local filtering is carried out by the unscented Kalman filter, whose recursion can be reviewed in Algorithm 3, Chapter 3.2.

Algorithm 9 C-CJPDAF

1: **function** C-CJPDAF($\hat{x}(0|0)$, $P(0|0)$)
2: **for all** time $k = 1, 2, \ldots$ **do**
3: $(\hat{x}(k|k-1), P(k|k-1)) \longleftarrow$ ***Local UKF Prediction***
4: $z_l \in m(k) \longleftarrow$ ***Local Measurement validation***
5: ***Local M/N logic***

6: **for all** unassociated measurements $\in Z_{NP}$ **do**

7: ***Local TPD gating***

8: **end for**

9: **for all** validated measurements $l = 1, \ldots, m(k)$ **do**

10: $\beta_l(k) \longleftarrow$ ***Local CJPDA***

11: $\nu_l(k) \longleftarrow z_l(k) - \hat{z}(k|k-1)$

12: **end for**

Local UKF Correction

13: $Z(k|k-1) \longleftarrow h(k, \mathcal{X}(k|k-1))$

14: $S(k) \longleftarrow Z(k|k-1)WZ(k|k-1)' + R(k)$

15: $T(k) \longleftarrow \mathcal{X}(k|k-1)WZ(k|k-1)'$

16: $K(k) \longleftarrow T(k)S(k)^{-1}$

17: $\hat{z}(k|k-1) \longleftarrow Z(k|k-1)w_m$

18: $\nu(k) \longleftarrow \sum_{l=1}^{m(k)} \beta_l(k)\nu_l(k)$

19: $\hat{x}(k|k) \longleftarrow \hat{x}(k|k) = \hat{x}(k|k-1) + K(k)\nu(k)$

20: $P^c(k|k) \longleftarrow P(k|k-1) - K(k)S(k)K(k)'$

21: $\tilde{P}(k) \longleftarrow K(k)\big[\sum_{l=0}^{m(k)} \beta_l(k)\nu_l(k)\nu_l(k)' - \nu(k)\nu(k)'\big]K(k)'$

22: $P(k|k) \longleftarrow \beta_0(k)P(k|k-1) + [1-\beta_0(k)]P^c(k|k) + \tilde{P}(k)$

Consensus

23: **for all** consensus steps $\ell = 0, 1, \ldots, L-1$ **do**

24: ***transmit***$\Big(\hat{x}^i(k|k,\ell),\ P^i(k|k,\ell)\Big)$

25: ***wait until*** $\forall j \in \mathcal{N}^i, \Big[\hat{x}^j(k|k,\ell),\ P^j(k|k,\ell)\Big]$ ***is received***

26: **for all** $i, j \in \mathcal{N}\ i \neq j$ **do**

27: $\pi^{i,j} \longleftarrow \dfrac{1}{\max(N_i, N_j)}$

28: **end for**

29: **for all** $i \in \mathcal{N}$ **do**

30: $\pi^{i,i} \longleftarrow 1 - \sum_{j \in \mathcal{N}^i\ j \neq i} \pi^{i,j}$

31: **end for**

Track-to-track association

CI fusion

$\Omega^i(k|k,\ell+1) \longleftarrow \sum_{j\in\mathcal{N}^i} \pi^{i,j}\,\Omega^j(k|k,\ell)$

$q^i(k|k,\ell+1) \longleftarrow \sum_{j\in\mathcal{N}^i} \pi^{i,j}\,q^j(k|k,\ell)$

end for

end for

return $\left[\Omega^i(k|k,L),\, q^i(k|k,L)\right]$

end function

8.4.2 Consensus CJPDAF with Gaussian Mixture implementation (CGM-CJPDAF)

The only differences between this algorithm and the CGM-CJPDAF are the following

- A Gaussian Mixture implementation for distributed fusion is used in order to reduce computational and communication costs. Specifically, the processed information in each node is expressed as a Gaussian mixture, defined in (8.49), whose Gaussian components are the posterior PDFs of each track. All the mixture weights α_j get the same value $1/N_G$, so that the constraint $\sum_{j=1}^{N_G} \alpha_j = 1$ is satisfied. Once the data has been exchanged with neighbors, Gaussian components of neighboring nodes are fused together.

- Track-to-track association is not performed as in Section 8.4.1. Instead, a gate (8.62) is set up in order to prevent the fusion between Gaussian components not originating from the same track. This gate simply compares the statistical distance between two Gaussian components with a given threshold.

CGM-CJPDAF can be also be described by the same steps $1, 2, 3, 4, 5$ as in C-CJPDAF. Then consensus takes place as follows:

6. ***Consensus.*** Each node i involves the subnetwork $\mathcal{N}^i$ in order to perform consensus. Specifically, the following operations take place:

   (a) ***Information exchange.*** Each node exchanges information (GM representation of the tracks) with neighboring nodes. Agent i transmits its data to node $j \in \mathcal{N}^j$ and receives information from $j \in \mathcal{N}^i$.

   (b) ***GM-GCI fusion.*** Gaussian components whose Mahalanobis distance falls inside the gate limited by a given fusion threshold, are fused together using the equations (8.58)-(8.61).

After the fusion, confirmed tracks are displayed on screen, and then they will start the local recursion for the following scan, together with unassociated tracks and any track received from the neighbors. As a matter of fact, Gaussian components representing unassociated confirmed tracks are kept locally for a certain number of time intervals.

In Algorithm 10, the pseudo-code summarizes the CGM-CJPDAF recursion, for each node and each track. Note that the term $s^i(x, \ell)$ denotes the Gaussian mixture corresponding to node i, $\ell - th$ consensus step and time k (which is not indicated).

Algorithm 10 CGM-CJPDAF

1: **function** CGM-CJPDAF($\hat{x}(0|0)$, $P(0|0)$)
2: **for all** time $k = 1, 2, \ldots$ **do**
3: $(\hat{x}(k|k-1), P(k|k-1)) \longleftarrow$ ***Local UKF Prediction***
4: $z_l \in m(k) \longleftarrow$ ***Local Measurement validation***
5: ***Local M/N logic***

6: **for all** unassociated measurements $\in Z_{NP}$ **do**

7: ***Local TPD gating***

8: **end for**

9: **for all** validated measurements $l = 1, \ldots, m(k)$ **do**

10: $\beta_l(k) \longleftarrow$ ***Local CJPDA***

11: $\nu_l(k) \longleftarrow z_l(k) - \hat{z}(k|k-1)$

12: **end for**

Local UKF Correction

13: $Z(k|k-1) \longleftarrow h(k, \mathcal{X}(k|k-1))$

14: $S(k) \longleftarrow Z(k|k-1)WZ(k|k-1)' + R(k)$

15: $T(k) \longleftarrow \mathcal{X}(k|k-1)WZ(k|k-1)'$

16: $K(k) \longleftarrow T(k)S(k)^{-1}$

17: $\hat{z}(k|k-1) \longleftarrow Z(k|k-1)w_m$

18: $\nu(k) \longleftarrow \sum_{l=1}^{m(k)} \beta_l(k)\nu_l(k)$

19: $\hat{x}(k|k) \longleftarrow \hat{x}(k|k) = \hat{x}(k|k-1) + K(k)\nu(k)$

20: $P^c(k|k) \longleftarrow P(k|k-1) - K(k)S(k)K(k)'$

21: $\tilde{P}(k) \longleftarrow K(k)\big[\sum_{l=0}^{m(k)} \beta_l(k)\nu_l(k)\nu_l(k)' - \nu(k)\nu(k)'\big]K(k)'$

22: $P(k|k) \longleftarrow \beta_0(k)P(k|k-1) + [1-\beta_0(k)]P^c(k|k) + \tilde{P}(k)$

Consensus

23: $s^i(x,\ell) \longleftarrow \Big(\hat{x}^i(k|k), P^i(k|k)\Big)$

24: **for all** consensus steps $\ell = 0, 1, \ldots, L-1$ **do**

25: ***transmit***$\Big(s^i(x,\ell)\Big)$

26: ***wait until*** $\forall j \in \mathcal{N}^i, \Big[s^j(x,\ell)\Big]$ ***is received***

27: **for all** $i, j \in \mathcal{N}$ $i \neq j$ **do**

28: $\pi^{i,j} \longleftarrow \dfrac{1}{\max(N_i, N_j)}$

29: **end for**

30: **for all** $i \in \mathcal{N}$ **do**

31: $\pi^{i,i} \longleftarrow 1 - \sum_{j \in \mathcal{N}^i \; j \neq i} \pi^{i,j}$

32: **end for**

33: $\bar{s}^i(x, \ell + 1) \longleftarrow$ ***GCI fusion***

34: **end for**

35: **end for**

36: **end function**

8.4.3 Distributed Track Initialization for Incomplete Measurements

In multitarget multisensor tracking, the sensor network can be formed for instance by range-only (TOA) and/or bearing-only (DOA) sensors which, as discussed in Section 5.2 and Section 5.3 respectively, are characterized by limited sensing capabilities. In particular, a single distance or angle measurement is an incomplete observation, since it does not fully specify a position in Cartesian coordinates, from which new tracks can be initialized. Thus, track initialization cannot be carried out as described in Section 6.1 for RADAR sensors, but an additional procedure is required before techniques as the two-point differencing (Section 6.1.1) can be applied to generate new tracks. When the measurement equation provides only distance or angle information, fusion is required among multiple nodes, in order to counteract the lack of target observability from a single node. In this section, the problem of track initialization in multitarget multisensor scenarios in presence of incomplete measurements is addressed.

The described approach can associate and fuse measurements from spatially distributed neighboring sensors in order to detect real targets within the surveillance region and initialize their tracks. A probabilistic grid representa-

tion is used [13, 32], which comes from the concept of occupancy grid, first introduced by Moravec and Elfes for robotic mapping [25]. This approach has been widely used for environment modeling in robotics due to the simplicity of its implementation [9]. The basic idea is to divide the measurement space into a grid of equally sized spatial cells and associate each measurement with the probability that the target is located in a particular cell of the grid. This method is then used to combine data from multiple sensor locations, by projecting measurements of a common quantity onto a two-dimensional probability mass grid. This is used to determine the most likely coordinates of the object being measured.

The sensor measurement of the quantity of interest can be modeled by a Gaussian probability mass function over the range of possible values. Subsequently, by adding the independent PDFs, sampled at common discrete intervals, of all measurements taken from a single sensor, a discrete probability distribution is obtained. Then, the PMFs of all neighboring sensors can be combined together, in order to form a joint discrete probability matrix. Finally the target localization problem is converted to a local maxima problem in the probabilistic grid. Using this method, distance and angle measurements from different sensors can be combined together to get the measurement of a target in a Cartesian coordinate system.

Consider a surveillance area of $A \times B\,[m]$. Choosing an appropriate dimension of the grid d, the surveillance region can be divided into $K_A \times K_B$ grids, where $K_A = A/d$, $K_B = B/d$. Thus each pair (a, b) represents a specific cell, whose center point is $C_{ab} = (C_{x_{ab}}, C_{y_{ab}})$. Note that the value of d is important because it directly impacts the accuracy and computational cost of combining multiple measurements.

Given a measurement z_m^i, the probability of being originated from a target

located in the center point of cell (a, b) is defined as follows

$$P_m^i(a,b) = \mathcal{N}(z_m^i; z_{ab}^i, R^i) \tag{8.63}$$

where (a, b) denotes both the cell and the corresponding center point C_{ab}, since we consider for each cell only the PDF evaluated in the center point. Note that z_{ab}^i denotes the virtual measurement associated to (a, b), i.e.

$$\begin{cases} z_{ab}^i = \sqrt{(C_{x_{ab}} - p_x^i)^2 + (C_{y_{ab}} - p_y^i)^2} & \text{TOA} \\ z_{ab}^i = \angle[(C_{x_{ab}} - p_x^i) + j(C_{y_{ab}} - p_y^i)] & \text{DOA} \end{cases} \tag{8.64}$$

where $p^i = (p_x^i, p_y^i)$ is the known position of sensor i.

The discrete probability matrix is formed by evaluating $P_m^i(a, b)$ at discrete points in each cell of the grid. Thus, given a measurement m of the i-th sensor, one has

$$F_m^i = \begin{cases} \dfrac{P_m^i(a,b)}{\sum_{r=1}^{K_A}\sum_{c=1}^{K_B} P_m^i(r,c)} & \forall (a,b) \\ 0 & \text{otherwise} \end{cases} \tag{8.65}$$

For each measurement of each sensor, using one of the (9.9), the probability (8.65) can be calculated. Thus, for each sensor i providing M^i measurements, the probability distribution of all measurements acquired by the i-th sensor is

$$F^i = \frac{P^i(a,b)}{\sum_{r=1}^{K_A}\sum_{c=1}^{K_B} P^i(r,c)} \tag{8.66}$$

where

$$P^i(a,b) = \sum_{m=1}^{M^i} P_m^i(a,b) \tag{8.67}$$

Then, in order to combine information from all neighboring nodes, a fusion step is carried out. Recalling the consensus problem (8.17), one has

$$\bar{P}(a,b) = \frac{[P^i(a,b)]^\omega \, [P^j(a,b)]^{1-\omega}}{\sum_{r=1}^{K_A}\sum_{c=1}^{K_B} \left[P^i(a,b)\right]^\omega \left[P^j(a,b)\right]^{1-\omega}} \qquad i,j \in \mathcal{N}, \quad \omega \in (0,1) \tag{8.68}$$

Finally, the regional average computed in node i is

$$\bar{P}^i(a,b) = \frac{\prod\limits_{j\in\mathcal{N}^i} \left[\bar{P}^j(a,b)\right]^{\omega^{i,j}}}{\sum\limits_{r=1}^{K_A}\sum\limits_{c=1}^{K_B}\prod\limits_{j\in\mathcal{N}^i}\left[\bar{P}^j(r,c)\right]^{\omega^{i,j}}} \tag{8.69}$$

where the consensus weights must fulfill the following conditions

$$\begin{cases} \omega^{i,j} \geq 0 & \forall i,\, j \in \mathcal{N} \\ \sum\limits_{j\in\mathcal{N}^i} \omega^{i,j} = 1 & \forall i \in \mathcal{N} \end{cases} \tag{8.70}$$

In conclusion, the target localization problem can be solved by finding the local maximums in the grid. As a matter of fact, peaks represent the estimated position of targets. Consequently, the number of peaks can be considered as the number of targets (under the assumption of no clutter). Clearly, in presence of clutter, an additional track confirmation logic, such as M/N, is needed for track confirmation, in order to reduce the number of false tracks initialized.

Chapter 9

Simulation Experiments

In order to evaluate the performance of the algorithms described in Chapters 7 and 8, simulation experiments have been carried out in realistic scenarios. The obtained results will be shown and discussed. In particular, the performance of the proposed distributed CGM-CJPDAF algorithm, presented in Section 8.4.2, is first compared with C-CJPDAF (Section 8.4.2). Next, it is compared with both centralized Sequential CJPDAF (S-CJPDAF, Section 7.4) and the single-sensor CJPDAF algorithm running locally (L-CJPDAF). Moreover, the performance of CGM-CJPDAF will be examined, in a network of TOA and DOA sensors. The evaluation is carried out by Monte Carlo simulations, so that any performance metric is averaged over a sufficiently high number N_{MC} of Monte Carlo runs relative to independent noise realizations. Tracking performance will be evaluated in terms of position and velocity error, track fragmentation and quickness in initializing new tracks. In addition, robustness with respect to uncertainties on the parameters of the scenario (detection probability P_D and false alarm probability P_{FA}) is examined. To this end, the evaluation metrics used in the following simulations are

- Root Mean Square Error (RMSE)
- Track continuity

- Average Track Confirmation Time (ATCT)

The position RMSE is calculated as follows

$$PRMSE(k) =$$

$$= \sqrt{\frac{1}{N_T\, N_S\, N_{MC}} \sum_{n=1}^{N_{MC}} \sum_{i=1}^{N} \sum_{j=1}^{N_T} \left(p_{xj}(k,n) - \hat{p}^i_{xj}(k|k,n)\right)^2 + \left(p^i_{yj}(k,n) - \hat{p}^i_{yj}(k|k,n)\right)^2} \tag{9.1}$$

where N_T is the number of targets, N_S is the number of nodes and N_{MC} is the number of Monte Carlo simulations, carried out for the same target trajectories but different, independently generated, clutter and measurement noise realizations. Clearly, $p_j(k,n)$ denotes the position of the true target j at scan k, during the n-th Monte Carlo run, whereas $\hat{p}^i_j(k|k,n)$ is the position of track j, estimated by sensor i. Similarly, the RMSE on velocity is

$$VRMSE(k) =$$

$$= \sqrt{\frac{1}{N_T\, N_S\, N_{MC}} \sum_{n=1}^{N_{MC}} \sum_{i=1}^{N} \sum_{j=1}^{N_T} \left(v_{xj}(k,n) - \hat{v}^i_{xj}(k|k,n)\right)^2 + \left(v^i_{yj}(k,n) - \hat{v}^i_{yj}(k|k,n)\right)^2} \tag{9.2}$$

RMSE evaluates the goodness of estimation once the target has been localized, but this metric is not useful to determine whether there are interruptions along the track. To this end, track continuity is computed. It accounts for the percentage of target trajectory which is really tracked, so that fragmented tracks will result in a low track continuity value. Both track continuity and RMSE are evaluated following the track-to-truth assignment method, discussed in [10] and [15]. The basic idea is that first a measure of fit (MOF) between confirmed tracks and truth is computed. Next, those pairings satisfying an acceptance criterion for the MOF, are placed in a global track-to-truth assignment matrix. Note that the assignment solution can be computed using methods such as the auction or the Hungarian algorithms, as mentioned in Section 8.2.3. The recommended MOF is the kinematic statistical distance

between the current track state estimate and truth, defined by

$$d^2 \triangleq \tilde{x}' P_{pv}^{-1} \tilde{x} \tag{9.3}$$

where $\tilde{x}$ is the vector of position and velocity differences and P_{pv} is the Kalman filter covariance matrix, containing position and velocity variances and covariances. Comparing the normalized distance (9.3) with a gate value γ_A, track-to-truth pairings are obtained. Thanks to this association, RMSE and track continuity can be calculated. In particular, position and velocity errors are calculated for each track-to-truth matching and track continuity is determined by giving a numerical score to each Monte Carlo run. A point is awarded at each time interval to each target for which the currently assigned track is the same as the one at the previous time. No points are awarded if there is no assigned track or if a switch occurred over the time interval. Therefore, track switching is penalized as well as tracking errors that are larger than those predicted by the covariance matrix. They cause a failure of the gate test that uses the distance defined in (9.3). As a matter of fact, failure of the gate test results in the loss of of two points since both the current time interval and the next are affected. The Measure Of Effectiveness (MOE) defined above can be easily converted to a correct tracking percentage, by computing the ratio between the total score and the possible score.

Once defined the metrics used for performance evaluation, let us now consider the multitarget tracking scenarios examined in the simulations. All the scenarios, listed below, are 2-dimensional (planar) and dynamic, so that different targets enter the surveillance area at different time instants:

1. ***Air surveillance:*** the scenario is characterized by the presence of three targets appearing in the surveillance region of $50 \times 50\,[km^2]$, wherein two radars are deployed.

2. ***Ground surveillance:*** a sensor network, formed by one bearing-only

(DOA) and two range-only (TOA) sensors, tracks three vehicular targets over a surveillance area of $5 \times 5\,[km^2]$.

In the following 2D multitarget simulations, the target state consists of Cartesian position and velocity components along the two axes x and y. The motion of each target is modeled according to the nearly-constant velocity model:

$$x(k+1) = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix} x(k) + w(k) \tag{9.4}$$

where T is the sampling interval and $w(k)$ is a zero-mean white process noise with covariance matrix

$$Q = \sigma_w^2 \begin{bmatrix} \frac{1}{4}T^4 & \frac{1}{2}T^3 & 0 & 0 \\ \frac{1}{2}T^3 & T^2 & 0 & 0 \\ 0 & 0 & \frac{1}{4}T^4 & \frac{1}{2}T^3 \\ 0 & 0 & \frac{1}{2}T^3 & T^2 \end{bmatrix} \tag{9.5}$$

where $\sigma_w = 2[m/s^2]$.

Sensors are characterized by non unity detection probability P_D and generate clutter, which is modeled as a Poisson Process with parameter λ (clutter density, given by (5.14)) and uniform spatial distribution over the surveillance region.

Scenario 1

In this first scenario there are two RADAR sensors linked in a geometric network, providing measurements of range and azimuth. The measurement equation is given by

$$z^i = \begin{bmatrix} r^i \\ \theta^i \end{bmatrix} + v^i = h^i(x) + v^i \tag{9.6}$$

where

$$v^i \sim wn(0, R^i) \tag{9.7}$$

and the measurement function is

$$h^i(x) = \begin{cases} \sqrt{(p_x - p_x^i)^2 + (p_y - p_y^i)^2} \\ \angle[(p_x - p_x^i) + j(p_y - p_y^i)] \end{cases} \tag{9.8}$$

p^i is the position of sensor i in Cartesian coordinates. For this scenario $p^1 = (0, 0); p^2 = (50000, 50000)$. Both range and azimuth errors are assumed independent, so that $R = diag\{\sigma_r^2, \sigma_\theta^2\}$, σ_r and σ_θ being the standard deviations of range and, respectively, azimuth measurement noise. Their values in this scenario are the same for both sensors, in particular $\sigma_r = 100\,[m]$ and $\sigma_\theta = 1\,[^\circ]$. Due to the nonlinearity of the sensor, UKF (Section 3.2) is used in each sensor.

In the scenario under consideration three targets with nearly-constant velocity motion appear in the surveillance area in different time instants, as shown in Figure 9.1. The duration of each simulation is fixed to 300 $[s]$ (100 samples).

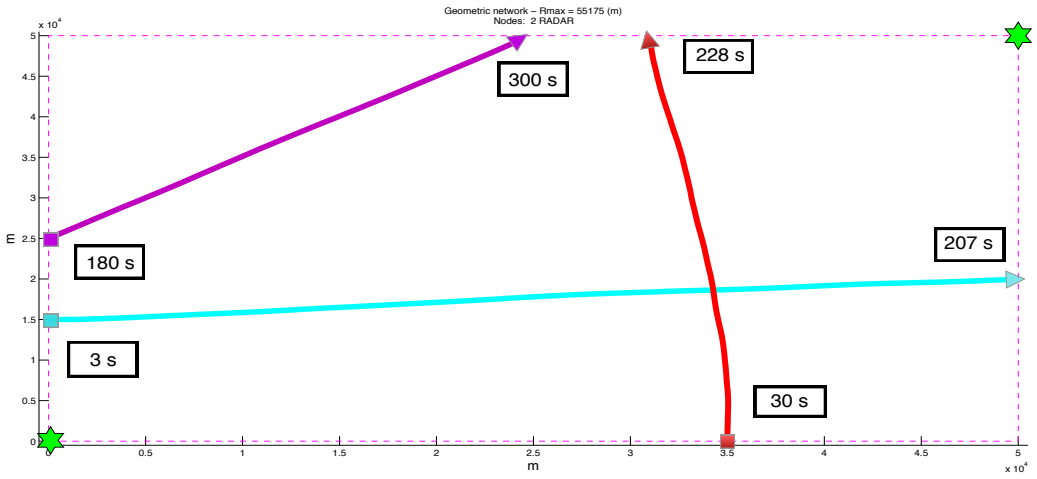

Figure 9.1: Target trajectories in Scenario 1.

The proposed algorithm CGM-CJPDAF (Section 8.4.2) is first compared to the analogous C-CJPDAF-T2TA (Section 8.4.1) which, before consensus takes place, carries out a track-to-track association procedure. The parameters used in simulations for Scenario 1 are reported in Table 9.1.

Parameter	**Description**	**Value**
P_D	probability of detection	0.9
P_G	gating probability	0.9997
λ	clutter density	5
M/N	confirmation logic	2/3
N_2	track deletion threshold	5
γ	validation gate threshold	16
T	sampling time	3 [s]
N_T	number of targets	3
N_S	number of nodes	2
N_{MC}	number of MC runs	300
σ_r	range measurement std. dev.	100 $[m]$
σ_θ	angle measurement std. dev.	1 [°]
σ_w	process noise std. dev.	2 $[m/s^2]$

Table 9.1: Parameters adopted in Scenario 1 simulations.

Multitarget tracking performance is evaluated in terms of position and velocity RMSE, track continuity and average time for track confirmation. Table 9.5 shows a comparison between CGM-CJPDAF and C-CJPDAF-T2TA in terms of track continuity and ATCT. Figure 9.2 and Figure 9.3 show position RMSE and velocity RMSE, respectively.

Statistics	C-CJPDAF-T2TA	CGM-CJPDAF
track continuity	0.887	0.868
ATCT	10.006	11.097

Table 9.2: CGM-CJPDAF vs C-CJPDAF-T2TA: Track continuity and ATCT.

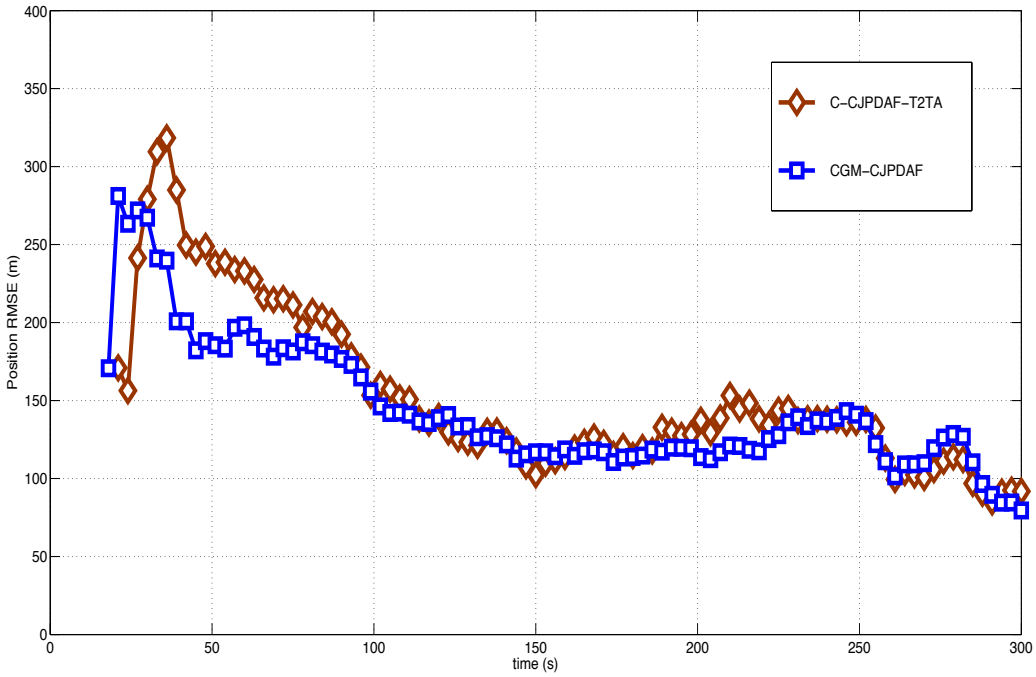

Figure 9.2: Comparison between CGM-CJPDAF and C-CJPDAF-T2TA: Position RMSE.

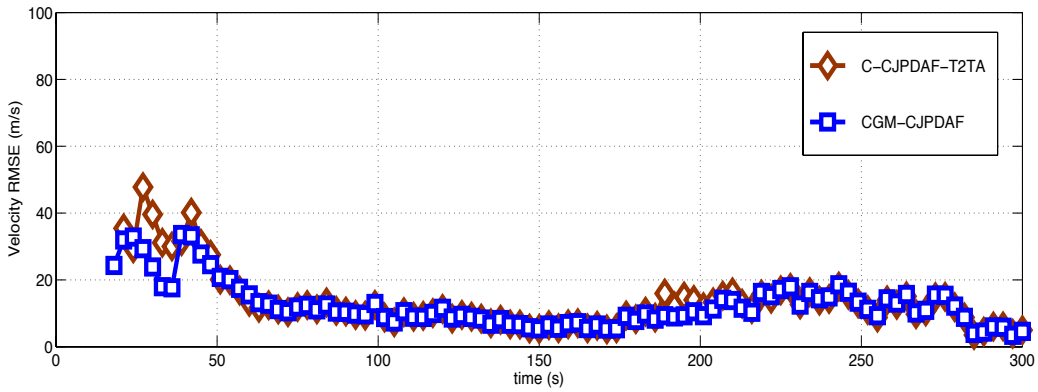

Figure 9.3: Comparison between CGM-CJPDAF and C-CJPDAF-T2TA: Velocity RMSE.

As shown in Figure 9.2 and Figure 9.3, the performance obtained with the two filters are very similar, as expected. This means that in the scenario under consideration both algorithms manage to "recognize" and thus fuse the same tracks obtained locally and received by the other sensor. Both filters perform a single consensus step.

Once verified that CGM-CJPDAF and C-CJPDAF-T2TA present analogous performance, CGM-CJPDAF is compared with the centralized Sequential CJPDAF (S-CJPDAF) and the single-sensor CJPDAF, in order to complete the evaluation. In Figure 9.4, Figure 9.5 and Table 9.3 the different metrics are illustrated for the three filters.

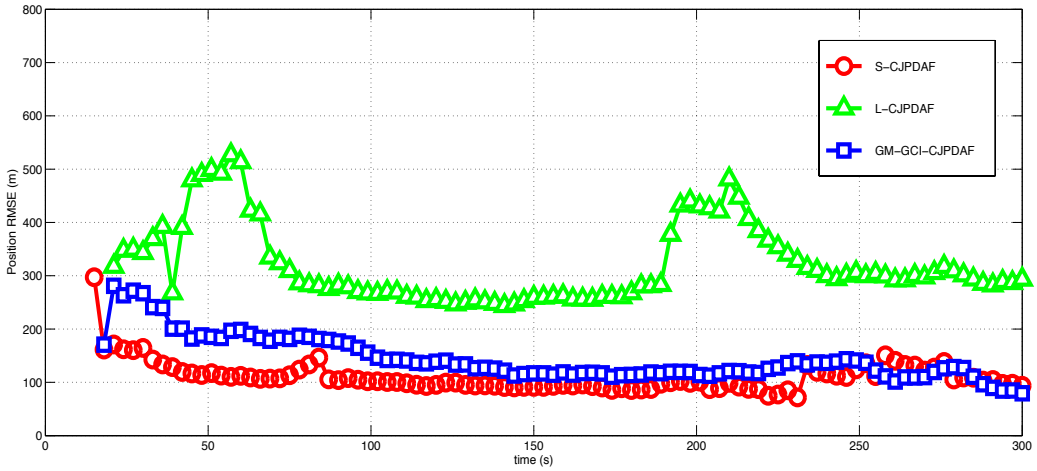

Figure 9.4: Comparison between CGM-CJPDAF, S-CJPDAF and L-CJPDAF: Position RMSE. $P_D = 0.9$.

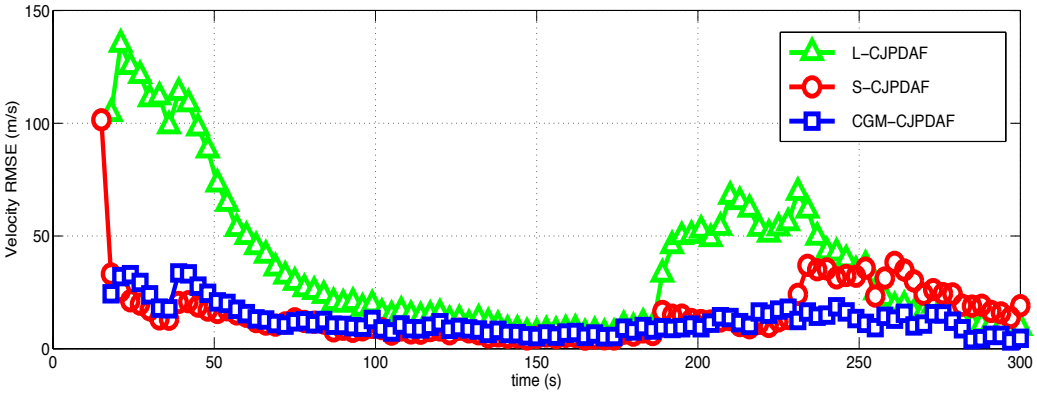

Figure 9.5: Comparison between CGM-CJPDAF, S-CJPDAF and L-CJPDAF: Velocity RMSE. $P_D = 0.9$.

These results show that, by applying consensus, performance of distributed algorithms is comparable to the one provided by the centralized tracker S-CJPDAF, which collects the set of measurements of two sensors in a central node. In addition, CGM-CJPDAF succeeds in improving the performance of L-CJPDAF, which is the algorithm carried out locally, in a single sensor.

Statistics	**L-CJPDAF**	**CGM-CJPDAF**	**S-CJPDAF**
PRMSE	320.084	144.405	132.385
track continuity	0.982	0.868	0.834
ATCT	5.867	11.097	6.303

Table 9.3: CGM-CJPDAF vs C-CJPDAF-T2TA: Track continuity and ATCT.

Note that, compared to L-CJPDAF and S-CJPDAF, CGM-CJPDAF is characterized by a higher average time for track confirmation.

Next, simulations have been carried out for target detection probability $P_D = 0.8$.

Parameter	**Description**	**Value**
P_D	probability of detection	0.8
P_G	gating probability	0.9997
λ	clutter density	5
M/N	confirmation logic	2/3
N_2	track deletion threshold	5
γ	validation gate threshold	16
T	sampling time	3 [s]
N_T	number of targets	3
N_S	number of nodes	2
N_{MC}	number of MC runs	300

Table 9.4: Parameters adopted in Scenario 1 simulations, $P_D = 0.8$.

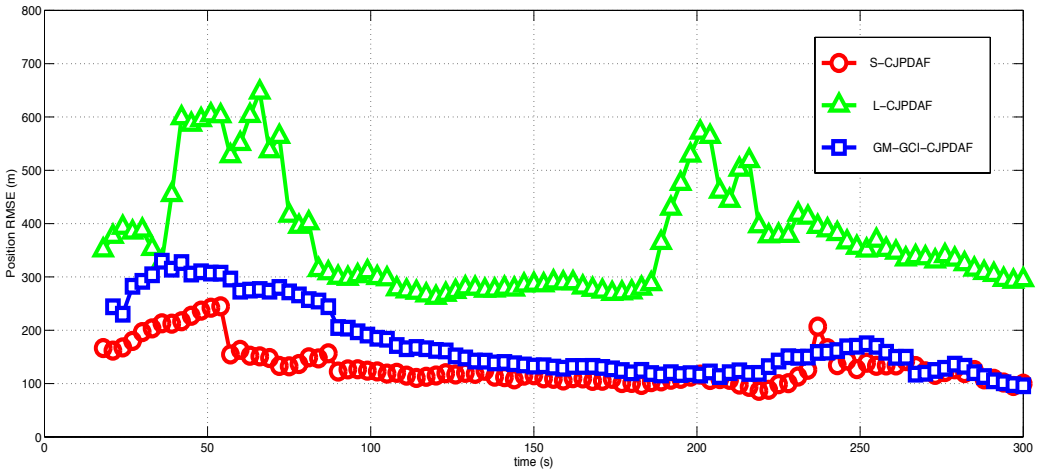

Figure 9.6: Comparison between CGM-CJPDAF, S-CJPDAF and L-CJPDAF: Position RMSE. $P_D = 0.8$.

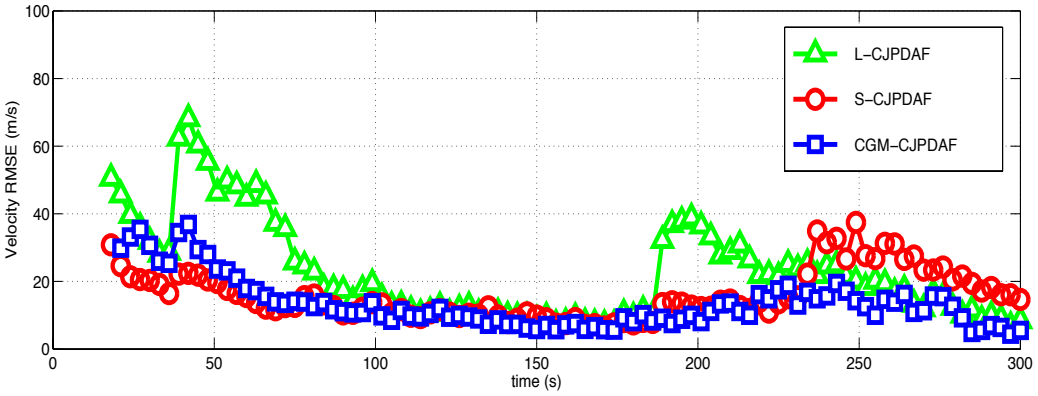

Figure 9.7: Comparison between CGM-CJPDAF, S-CJPDAF and L-CJPDAF: Velocity RMSE. $P_D = 0.8$.

Statistics	**L-CJPDAF**	**CGM-CJPDAF**	**S-CJPDAF**
track continuity	0.951	0.819	0.760
ATCT	6.200	13.654	9.431

Table 9.5: CGM-CJPDAF vs C-CJPDAF-T2TA: Track continuity and ATCT. $P_D = 0.8$.

Scenario 2

In this scenario the sensor network is formed by one bearing-only and two range-only sensors, characterized by the following measurement functions:

$$\begin{cases} h^i(x) = \sqrt{(p_x - p_x^i)^2 + (p_y - p_y^i)^2} & \text{TOA} \\ h^i(x) = \angle[(p_x - p_x^i) + j(p_y - p_y^i)] & \text{DOA} \end{cases} \tag{9.9}$$

The standard deviation of DOA and TOA measurement noises are taken respectively as $\sigma_\theta = 1[^\circ]$ and $\sigma_r = 10[m]$.

The scenario under consideration is depicted in Figure 9.8. Three targets with nearly-constant velocity motion appear in the surveillance area of $5 \times 5\,[km^2]$ in different time instants. The sampling interval is $T = 5\,[s]$. The duration of each simulation is fixed to 400 $[s]$ (80 samples).

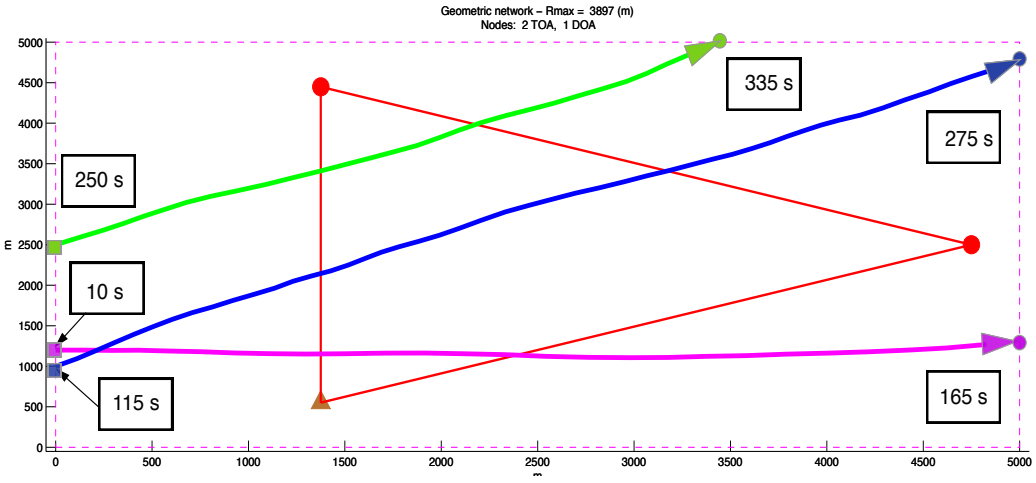

Figure 9.8: Target trajectories in Scenario 2.

For this scenario distributed track initialization based on probability grids, described in Section 8.4.3, has been implemented, due to the fact that TOA and DOA sensors provide incomplete measurements i.e. the single node lacks of target observability. The parameters used in this simulation are reported in Table 9.6. The grid has been divided into 400 cells of $250 \times 250\,[m^2]$.

Multitarget tracking performance is evaluated in terms of position and velocity RMSE, averaged over $N_{MC} = 100$ Monte Carlo runs for the same target trajectories but different, independently generated, clutter and measurement noise realizations.

Parameter	**Description**	**Value**
P_D	probability of detection	0.99
P_G	gating probability	0.9997
λ	clutter density	1
M/N	confirmation logic	2/3
N_2	track deletion threshold	2
γ	validation gate threshold	16
T	sampling time	5 [s]
N_T	number of targets	3
N_S	number of nodes	3
N_{MC}	number of MC runs	100
σ_r	range measurement std. dev.	10 $[m]$
σ_θ	angle measurement std. dev.	1 [°]
σ_w	process noise std. dev.	2 $[m/s^2]$

Table 9.6: Parameters adopted in Scenario 2 simulations.

As shown in Figure 9.9 and Figure 9.10, the algorithm implementing a probability grid approach for track initialization, confirms good performance in a low-density clutter scenario.

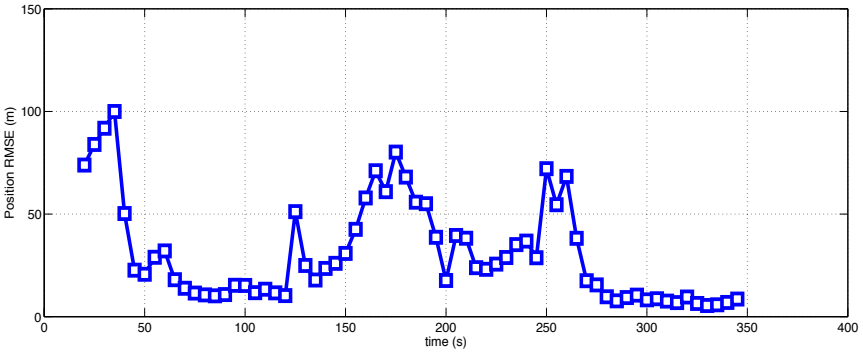

Figure 9.9: Probability Grid Initialization: Position RMSE.

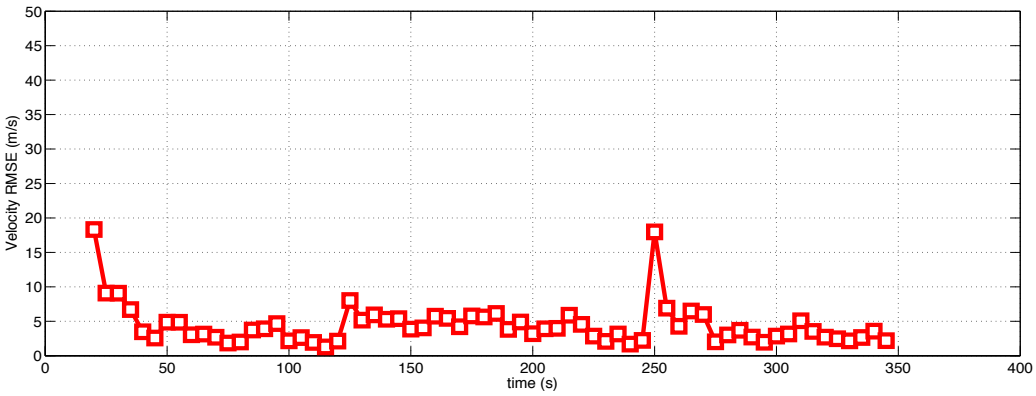

Figure 9.10: Probability Grid Initialization: Velocity RMSE.

Chapter 10

Conclusions

This thesis has considered the multitarget multisensor tracking problem. This essentially concerns the joint detection and estimation of the unknown and time-varying number of targets in the scenario, as well as the estimation of the kinematic state of each target, given a sequence of observation sets. Multitarget multisensor tracking problems can be tackled in essentially two ways:

- by centralized architectures, in which a central data processor receives all the measurements;
- by distributed approaches that carry out scalable processing without coordination of a central unity.

The focus of this dissertation has been on distributed multitarget tracking of multiple targets over a sensor network. In particular, consensus algorithms, capable of providing fully distributed and scalable fusion of the information collected from multiple sensors in presence of clutter and noisy observations, were considered. A comparison of both C-CJPDAF and CGM-CJPDAF with a sequential centralized filter, has shown that the two methods developed in this thesis give comparable performance to conventional methods of multiple

target tracking using a central data processor. Furthermore, it is shown that the proposed distributed multitarget trackers can successfully track the correct targets in reasonably high levels of clutter.

In order to solve the distributed multitarget tracking problem, several issues have been encountered. These sub-problems have been progressively addressed throughout the whole thesis and their solutions have been subsequently combined in the proposed distributed algorithms.

First, multitarget tracking has been considered, which is essentially the combination of data association and estimation. Due to the employment of nonlinear sensors, discussed in Chapter 5, recursive nonlinear estimation (Chapter 3) is needed. Specifically, the Unscented Kalman Filter (UKF) has been implemented for target state estimation. Regarding data association, Probabilistic and hard-decision single-scan techniques have been described throughout Chapter 6. Particular interest has been given to Bayesian approaches for multiple target data association, which is carried out in each node of the network. To this end, an approximated version of Joint Probabilistic Data Association (CJPDAF), that reduces computational burden, has been used in the local trackers.

Next, the resulting data processing in each node has been further developed in order to extend the multitarget tracking problem to a multisensor environment. Consensus has been implemented, i.e. the information between neighboring nodes is collected in each sensor, and a distributed fusion, which guarantees scalability, takes place. Furthermore, a new method for track initialization, based on probability grids, has been proposed for distributed multitarget problems in which sensor networks are characterized by limited sensing capabilities. In conclusion, simulation results are provided in Chapter 9 in order to demonstrate the effectiveness of the proposed distributed multitarget tracking algorithms for tracking multiple targets over sensor networks.

Possible topics for future work are:

- Extension to sensors with different field of view (FOV);
- Further investigations on track initialization for incomplete measurements;
- Comparison with distributed CPHD filtering techniques [5].

Bibliography

[1] Y. Bar-Shalom and T.E. Fortmann. *Tracking and Data Association*. Mathematics in Science and Engineering Series. Acad. Press, 1988.

[2] Y. Bar-Shalom and X.R. Li. Multitarget-multisensor tracking: principles and techniques. 1995.

[3] Y. Bar-Shalom, X.R. Li, and T. Kirubarajan. *Estimation with Applications to Tracking and Navigation: Theory Algorithms and Software*. Wiley, 2004.

[4] Y. Bar-Shalom and E. Tse. Tracking in a cluttered environment with probabilistic data association. *Automatica*, pages 451–460, 1975.

[5] G. Battistelli, L. Chisci, C. Fantacci, A. Farina, and A. Graziano. Consensus cphd filter for distributed multitarget tracking, 2013.

[6] C. Bettstetter. On the minimum node degree and connectivity of a wireless multihop network. *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, pages 80–91, 2002.

[7] C. Bettstetter. On the connectivity of ad hoc networks. *The Computer Journal*, pages 432–447, 2004.

[8] C. Bettstetter and C. Hartmann. Connectivity of wireless multihop networks in a shadow fading environment. pages 571–579, 2005.

[9] A. Birk and S. Carpin. Merging occupancy grid maps from multiple robots. *Proceedings of the IEEE*, pages 1384–1397, 2006.

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

[11] G. C. Calafiore and F. Abrate. Distributed linear estimation over sensor networks. *International Journal of Control*, pages 868–882, 2009.

[12] L. Chisci, G. Battistelli, F. Papi, and S. Marrocchi. An information-theoretic approach to distributed state estimation. *IFAC World Congress 2011*, 2011.

[13] D. Elsaesser. Sensor data fusion using a probability density grid. In *Information Fusion, 2007 10th International Conference on*, pages 1–8, 2007.

[14] R. J. Fitzgerald. Development of practical pda logic for multitarget tracking by microprocessor. pages 889–898, 1986.

[15] B. E. Fridling and O. E. Drummond. Performance evaluation methods for multiple-target-tracking algorithms. pages 371–383, 1991.

[16] J. Hartikainen and S. Särkkä. Optimal filtering with kalman filters and smoothers–a manual for matlab toolbox ekf/ukf. *Helsinki University of Technology, Espoo, Finland*, 2008.

[17] S. Haykin. *Kalman Filtering and Neural Networks*. Adaptive and Learning Systems for Signal Processing, Communications and Control Series. Wiley, 2004.

[18] S. J. Julier and J. K. Uhlmann. A new extension of the kalman filter to nonlinear systems. pages 182–193, 1997.

[19] S. J. Julier and J. K. Uhlmann. A non-divergent estimation algorithm in the presence of unknown correlations. In *American Control Conference*, volume 4, pages 2369 –2373, jun 1997.

[20] S.J. Julier and J.K. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92:401–422, 2004.

[21] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[22] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, pages 83–97, 1955.

[23] S. Boyd L. Xiao and S. Lall. A scheme for robust distributed sensor fusion based on average consensus. In *Proceedings of the International Conference on Information Processing in Sensor Networks*, pages 63–70, 2005.

[24] R. P. S. Mahler. Optimal/robust distributed data fusion: a unified approach. pages 128–138, 2000.

[25] H.P. Moravec and A. Elfes. High resolution maps from wide angle sonar. In *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*, volume 2, pages 116–121, 1985.

[26] J. Munkres. Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics*, pages 32–38, 1957.

[27] L. Y. Pao and C. W. Frei. A comparison of parallel and sequential implementations of a multisensor multitarget tracking algorithm, 1995.

[28] C.H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: algorithms and complexity*. Dover Publications, Incorporated, 1998.

[29] R.A. Singer. Estimating optimal tracking filter performance for manned maneuvering targets. *Aerospace and Electronic Systems, IEEE Transactions on*, pages 473–483, 1970.

[30] E.A. Wan and R. Van der Merwe. The unscented kalman filter for nonlinear estimation. In *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*, pages 153–158.

[31] D. Willner, C.B. Chang, and K. P. Dunn. Kalman filter algorithms for a multi-sensor system. In *Decision and Control including the 15th Symposium on Adaptive Processes, 1976 IEEE Conference on*, volume 15, pages 570–574, 1976.

[32] S. Xiao Z. Zhao, X. Wang and D. Dai. Grid-based probability density matrix for multi-sensor data fusion. In *Microelectronics Electronics, 2009. PrimeAsia 2009. Asia Pacific Conference on Postgraduate Research in*, pages 205–208, 2009.