ICT Call 4
RoboEarth Project
2010-248942

Deliverable D2.1:

Data Association and Tracking: A Literature Survey

April 1, 2010

Project acronym:	RoboEarth
Project full title:	Robots sharing a knowledge base for world modeling and learning of actions
Work Package:	2
Document number:	D2.1
Document title:	Data Association and Tracking: A Literature Survey
Version:	1
Delivery data:	April 1, 2010
Nature:	Report
Dissemination level:	Public
Authors:	Jos Elfring (TU/e) Rob Janssen (TU/e) René van de Molengraft (TU/e)

The research leading to these results has received funding from the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement n°248942 RoboEarth.

Contents

Summary

Part of the objective of work package two in the RoboEarth project is to develop a fast, real-time capable multi-sensor multi-target data association and objects tracking approach that works in unstructured and dynamically changing environments with typically a large number of objects. Such an algorithm allows the robot to safely navigate through the hospital room and to react on moving object. This literature survey gives an extensive overview of the recent advances in data association and tracking approaches.

Chapter 1

Introduction

In the RoboEarth project, work package 2 (WP2) concerns world modeling. This work package includes the development of standardized object definitions and descriptions that allow exchange between robots and RoboEarth. Another important task in WP2 is the development of a fast real-time capable multi-sensor multi-target data association and tracking approach that works in both structured and unstructured environments. This approach will be an important part of the local world model on each robot, which allows a robot to plan his navigation tasks properly and to avoid obstacles. Apart from this local world model, a global world model in RoboEarth, that provides the local world model with initial guesses, has to be maintained.
This literature survey focusses on data association and target tracking, part of the local world model on the robot. Tracking targets involves many difficulties, *e.g.*, association of measurements with objects, how many targets have to be tracked, and how do these targets move. This report investigates the state of the art in multi-sensor multi-target tracking. An extensive overview of existing methods is presented and each of these methods is briefly explained. Furthermore, relevant references are given further explaining the algorithms. Whenever found useful, appendices are used to explain details.
The report starts with Chapter 2, where the problem is introduced. Chapter 3 explains the state of the art regarding data association algorithms. The main focus of these algorithms is associating measurements with objects and the distinction between correct and false measurements. Each section explains a data association method and addresses its strong and weak points. Then, Chapter 4 continues with the state of the art (multi-target) tracking algorithms. These algorithms, mainly focus on approximating the location and velocity of the objects based on the outcome of the data association algorithms. Again, each method is commented on. In Chapter 5 conclusions and recommendations regarding future research are given.

Chapter 2

Problem description

This document requires some knowledge about probability theory and for that reason, in Appendices A.1 and A.2, basic probability theory is explained. The system that is considered throughout this report is written

$$x_k = f_k(x_{k-1}, u_{k-1}, v_{k-1}), \tag{2.1}$$

where $\{x_k,\ k \in \mathbb{N}\}$ is the discrete time state sequence, f_k is a possibly nonlinear and usually time dependent function, u_{k-1} is the (known) control input which is not necessarily present, and $\{v_{k-1},\ k \in \mathbb{N}\}$ is an independent and identically distributed (i.i.d.) process noise sequence, n_x, n_u, and n_v are the dimensions of the state, input, and process noise vectors and $\mathbb{N}$ is the set of natural numbers. The theory described throughout this report can equally well be used for continuous time systems. The discrete time definitions are in line with the notation in, *e.g.*, Arulampalam *et al.* [2002]. Usually, Equation (2.1) is assumed to be a Markov process, *i.e.*, x_{k-1} contains all measurement information y_{k-1} up to time $k-1$. The function f in Equation (2.1) can be obtained in various ways. If a physical model is used, the equation usually is written in state space form and the model is called white. If on the other hand a fixed model structure is used, *e.g.*, a neural network as explained in Appendix A.3 or Bishop [2006], the model is called black and only the input output behavior is considered relevant. In between are grey models, which are a mixture of both types of models.
In tracking, the goal will be to recursively estimate the states in Equation (2.1). The state sequence is assumed to be stochastic and, therefore, it is tried to find a probability density function (pdf) of the target states. In the robotic environment, the states typically represent positions and velocities of objects, but most of the algorithms work equally well in another context. In order to estimate the states, the following measurements are available:

$$z_k = h_k(x_k,\ w_k), \tag{2.2}$$

where h_k: $\mathbb{R}^{n_x} \times \mathbb{R}^{n_w} \to \mathbb{R}^{n_z}$ is a possibly nonlinear function, $\{n_k,\ k \in \mathbb{N}\}$ is an i.i.d. measurement noise sequence, and n_z and n_w are the dimensions of the measurement and measurement noise vectors z_k and w_k.

Chapter 3

Data association

3.1 The data association problem and measurement gates

Now that the the problem has been described, the focus is on the data association problem. While tracking objects, usually multiple measurements appear, *e.g.*, both due to targets and measurement noise. The incorrect measurements are referred to as false measurements or clutter. Data association deals with the problem of selecting the measurement that most probable is originated from the object to be tracked. If the wrong measurement is selected, or if the correct measurement is not detected at all, poor state estimates could be the result. A common first step in solving the data association problem is the selection of a validation region, sometimes called (measurement) gate. An algorithm defining a gate is described in Bar-Shalom & Fortmann [1988] and summarized in this section.
The gate is a region in which the next measurement is highly probable to appear. In order to define such a region, the target is assumed to be on a track, such that a predicted measurement $\hat{z}_{k|k-1}$ and the measurement prediction covariance matrix S_k are available (obviously the assumptions that are used to calculate the predicted measurement and the covariance matrix hold, see Appendix C.1. Now the gate is defined

$$\tilde{V}_k(\gamma) = \{z : \ \nu_k^T S_k^{-1} \nu_k \leq \gamma\}, \tag{3.1}$$

where the innovation ν and covariance matrix S_k are as in Appendix C.1, and γ is used to obtain a desired probability. In Bar-Shalom & Fortmann [1988], it is assumed that the true measurement at time k, conditioned on the old measurements up to time $k-1$, is normally distributed. Therefore, this γ can be obtained from tables of the χ^2-distribution given the dimension of the measurement n_z and a probability P_G (selected beforehand) that the new measurement will fall in the gate. From now on, all measurements that fall within the gate are called validated measurements. Validated measurements are denoted:

$$Z(k) \triangleq \{z_i(k)\}_{i=1}^{m_k}, \tag{3.2}$$

where the integer number m_k, with $m_k \geq 0$, is the number of validated measurements at time k. The problem of single target data association with $m_k = 4$ is summarized in Figure 3.1, where the size and shape of the ellipse are determined by S_k.
In reality, the number of objects to be tracked might exceed one so that it has to be decided which measurement is originated from which target. It becomes even more complex after realizing that the number of objects to be tracked usually is unknown and nonconstant, *i.e.*,

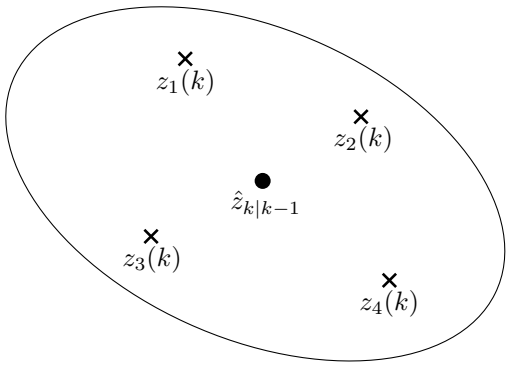

Figure 3.1: Data association problem: a predicted single target measurement with four validated measurements.

new objects can appear and disappear any time and anywhere which means that invalidated measurements can be relevant. So, a data association algorithm has to determine whether a measurement is correct or incorrect and if a new track has to be initialized or an existing track should be continued. The more complex situation is summarized in Figure 3.2, where $\hat{z}^1_{k|k-1}$ and $\hat{z}^2_{k|k-1}$ are the predicted measurements for track one and two respectively, $z_1(k)$, $z_2(k)$, $z_3(k)$ and $z_4(k)$ are validated measurements for the first track, $z_4(k)$, $z_5(k)$, $z_7(k)$ and $z_8(k)$ are validated measurements for the second track and all measurements that are not used for the continuation of a existing track can either be the result of clutter or newly appeared objects. Note that the possibility that none of the measurements belongs to the track one or two should be kept open, *i.e.*, one of the objects disappeared. Data association algorithms based on gates, clearly should include a strategy that is able to deal with the appearance and the disappearance of tracks.

3.2 Nearest neighbor standard filter

The nearest neighbor algorithm probably is the most straightforward approach solving the data association problem. This section briefly describes the nearest neighbor standard filter (NNSF), but many variant exist, *e.g.*, probabilistic nearest neighbor, distributed sequential nearest neighbor, suboptimal nearest neighbor, and global nearest neighbor (GNN), used in, *e.g*, Song *et al.* [2005], Zhang *et al.* [1996], and Leung *et al.* [1999]. Some details about the NNSF can be found in Bar-Shalom & Fortmann [1988].
The idea of the NNSF is to select the measurement that is closest to the predicted measurement $\hat{z}_{k|k-1}$, *i.e.*, it is an optimal solution in the sense that it minimizes the distance between predicted and measured points. In order to do so, for each of the measurements the distance d^2 is calculated using

$$d^2 = \nu_k^T S_k^{-1} \nu_k, \tag{3.3}$$

and the measurement with the shortest distance d^2 is believed to be the correct one. In Figure 3.2, this would result in the selection of measurements z_2 and z_5 for the first and second track respectively. This filter can be implemented for tracking any number of known tracks and an advantage is the low computational complexity. An obvious drawback is that, with some probability, the nearest neighbor is not the correct measurement. When the

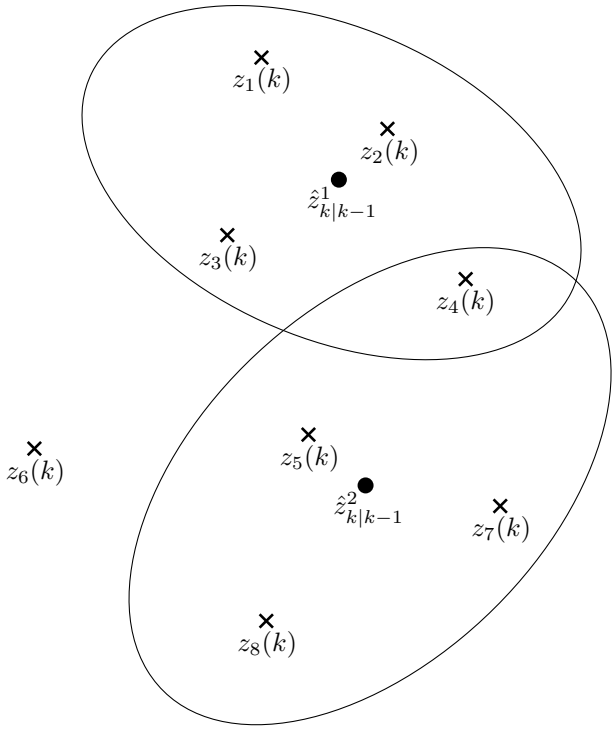

Figure 3.2: Association problem: two predicted target measurements with various (possibly validated) measurements.

number of false measurements increases, the data association problem is solved non-optimal more often and the performance of the NNSF degrades rapidly. Furthermore, one measurement can be associated with multiple tracks resulting in nonphysiological tracks. If the constraint that one measurement can only be associated with one track is added, then the GNN approach is obtained. More details about the GNN approach can be found in Blackman [2004], where the performance of such an approach is said to be rather poor. This filter is not able to deal with the appearance or disappearance of objects. As a result, it can only be used combined with a target birth/death model and it is not found to be a promising algorithm within complex situations.

3.3 Probabilistic data association filter

The probabilistic data association filter (PDAF) is a suboptimal algorithm that utilizes association probabilities for the latest measurements. The key idea of the filter is that a weighted average of all validated measurements, where probabilities are used as weights, is input for the tracking algorithm. The basic assumption is that the state is normally distributed according to the latest state estimate and covariance matrix. Furthermore, only one target is modeled whose track has been initialized and linear dynamic and measurement models are considered. These assumptions allow using the linear Kalman filter (LK), which plays a central role in the filter. In case of a nonlinear model, a linearization is recommended. A detailed explanation of this filter can be found in Bar-Shalom & Fortmann [1988] and in Appendix B.1.
The linearity assumption and the fact that only one initialized track is considered, are not always realistic. However, this filter has proven its value, see *e.g.*, Kirubarajan & Bar-Shalom [2004] and Abolmaesumi & Sirouspour [2004] . Furthermore, it is used as a starting point for improved techniques, as will be explained later on.

3.4 Joint probabilistic data association filter

The joint probabilistic data association filter (JPDAF) is an extension to the PDAF, that was introduced in Section 3.3, in the sense that it enables tracking multiple targets in clutter. More specifically, the difference is that the measurement to target association probabilities are calculated jointly across targets. Again, only the last measurement is used and each target is assumed to have its own linear dynamic and measurement model. Furthermore, the number of targets is assumed to be known and the result is a set of weighted averages. Again, the Kalman filter plays a key role. For more details is referred to Bar-Shalom & Fortmann [1988] or Appendix B.2.
An obvious disadvantage of this method is that it only considers linear models. In order to deal with nonlinear systems, linearizations can be used, however, this does only work if the nonlinearities are weak in the region of interest. Another disadvantage is the (key) assumption that the number of targets is known and constant. Despite these drawbacks, the JPDAF proved its value in, *e.g.*, Gorji *et al.* [2007], and it is a major improvement compared to the PDAF. Furthermore, this filter is able to deal with multiple sensors. However, both the PDAF and the JPDAF have the inability to recover from errors caused by, *e.g.*, erroneous matches that can be a result of noisy measurements.

In Schultz *et al.* [2003], a sample-based version of the JPDAF, called SJPDAF, is introduced. The Kalman filter is replaced by a particle filter, which allows dropping the Kalman related assumptions of Gaussian distributions and linear system dynamics. The particle filter is explained in Appendix C.2. In addition, Schultz *et al.* [2003] introduce a distribution over the number of objects, allowing a varying and unknown number of objects to be tracked. A generalization to multiple sensors and arbitrary proposal distributions is introduced in Vermaak *et al.* [2005] and is called Monte Carlo JPDAF (MC-JPDAF). Unfortunately, the MC-JPDAF assumes a fixed and known number of targets.

3.5 The track splitting filter

The PDAF and JPDAF are methods solving the data association problem recursively, more specifically, methods that only use one scan back measurements and states at time $k-1$. The track splitting filter described in this section is a batch method, *i.e.*, it uses a sequence of data obtained at multiple time instants. The track splitting filter, introduced by Smith & Buechler [1975], is a non-Bayesian algorithm that is abe to deal with an unknown number of targets. The main assumptions are linear dynamic and measurement models and Gaussian process and measurement noise. After the initialization, at time $k = 1$, the track is split up into m_k tracks, one for each validated measurement. Furthermore, for each track, a Kalman filter (explained in Appendix C.1) is implemented. Then m_k validation regions are calculated and at time $k = 2$, the procedure is repeated. If two measurements at successive times are close to each other and not associated, this could be used to initiate a new track. Clearly, this strategy has to deal with a rapidly increasing number of tracks and for that reason, likelihoods of all tracks are calculated. If the likelihood is lower than a predefined threshold, it will be eliminated. Note that this threshold is not equal to the probability that a sequence is correct.
Once a track has a long history, the likelihood of the track will be dominated by this long history. As a result, the response will be slow and the memory and computation requirements will rapidly increase. This a major drawback and, therefore, a sliding window can be included. Such a window ensures that only the last measurements are taken into account. Another drawback is that the track splitting filter allows shared measurements between tracks. This may result in nonphysiological tracks. A solution to this last problem, under the assumption of unity probability of detection, is given in Bar-Shalom & Fortmann [1988] and named "a joint likelihood method for track formation". This extension adds constraints to the track splitting filter. With these constraints only the feasible tracks are considered. A track is feasible if each measurement belongs to some track and each measurement belongs to only one track. Then, from all feasible tracks, the one with the highest joint likelihood is used to determine the trajectories of the targets.

3.6 Multiple hypothesis tracking

An approach that has some overlap with the track splitting filter from Section 3.5 is the multiple hypothesis filter (MHF), sometimes called multi-hypothesis tracker (MHT). The overlap lies in the fact that both (batch) methods postpone the data association decision until more information is available. It was introduced in Reid [1979] and an efficient implementation can be found in, *e.g.*, Cox & Hingorani [1996]. It is one of the most widely

used data association algorithms and this section briefly explains the MHF by means of an example.
If Figure 3.2 is considered, then a few feasible hypothesis can be made. An example of such a hypothesis could be:

> H_1: Track 1 is continued by z_2, track 2 is continued by z_5, a new track is initiated from z_6, and the other measurements are false alarms.

A less probable, but still valid hypothesis is:

> H_2: Track 1 and track 2 are not continued, z_4 and z_5 originate from new targets, and the other measurements are false alarms.

The first step of the MHF is the formulation of all feasible hypotheses. Then, when new data comes available, each hypothesis is expanded into a set of new hypotheses. This way a tree of hypothesis can be generated. With the formulation of each new hypothesis the compatibility constraint is maintained, *i.e.*, only feasible hypothesis are considered. The track score is used to assess the validity of the track. Details about its computation can be found in Blackman [2004]. As more data is measured, the size of the hypothesis tree can grow exponentially. To keep the size of the tree limited several techniques are described in Blackman [2004] and Reid [1979]. This includes:

- A first way to limit the computational and memory requirements is clustering. This means that the entire set of measurements is subdivided into sets of independent clusters. Each measurement that cannot be associated with an existing cluster is given a new cluster. This way the large problem is decomposed into a set of smaller problems allowing a more efficient implementation.
- A second way is by track and hypothesis pruning. In this method, at each time step, a part of the hypothesis tree is deleted. In order to explain this method, first some more details about the hypothesis tree are desired. Within a hypothesis tree, various families exist. One family contains all tracks that have the same "root node" (root measurement). This means that the various tracks within one family are not compatible. Clearly, there can exists multiple families at the same time. At time k, the pruning mechanism goes back N scans and selects new root nodes by picking the best hypothesis. This way, a part of the hypothesis tree is eliminated. Figure 3.3 gives an example similar to the example in Blackman [2004], where the numbers represent track numbers and two families, F_1 and F_2, are shown. Here $N = 2$, so the pruning algorithm goes back two steps in time. At time $k - 3$ a root node was selected, but new measurements are available at time k and new root nodes should be selected at time $k - 2$. In this example, the new root node is track 2, meaning that the left branch of the biggest tree will be pruned (gray circles). The above mentioned implementation by Cox & Hingorani [1996] uses a similar approach.

A MHT that grows linearly with the number of states and hence has lower computational costs than then the standard MHT is the probabilistic MHT (PMHT). It does not enumerate all possible feasible combinations but tries to find maximum likelihood solutions using an expectation-maximization (EM) algorithm instead. Bishop [2006] gives a detailed explanation of the EM algorithm and Streit [2006] explains the PMHT. In Willett *et al.* [2002], ten different PMHT implementations are compared and their strong and weak points

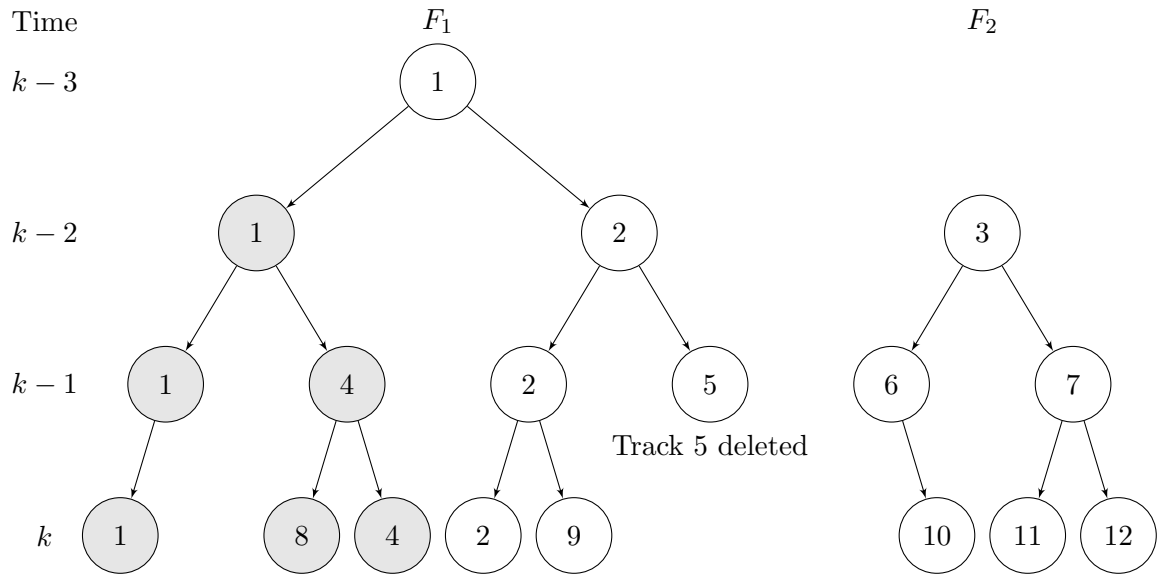

Figure 3.3: The family structure with 2 step back pruning.

are addressed. The performance is compared to a standard PDAF. The computational costs of the PDAF are lower and if tracks can be lost the performance of the PDAF is better. When the environment is moderate, *i.e.*, tracks are not likely to be lost, the PMHT is more accurate, according to Willett *et al.* [2002]. Despite this tempered optimism, others are reporting very successful implementations, *e.g*, Hempel & Pacheco [2009], where the PMHT was tested on active sonar sea trial data.

3.7 Viterbi data association

One of the key problems in hidden Markov models (HMMs) is the question "given an observation sequence and a HMM, how can the optimal (in some meaningful sense) state sequence be found?", see Rabiner [1989]. A common way to solve this problem is applying the Viterbi algorithm (VA), introduced by Viterbi [1967]. If the sequence of measurements is set onto a trellis, this VA can be used to solve the data association problem by finding an optimal sequence, *i.e.*, an optimal track. In order to be able to find any optimal solution, some cost function has to be defined. The cost function that is used for VA data association can be based on similar ideas as the NNSF, explained in Section 3.2. Using the VA in order to solve the data association problem results in the Viterbi data association (VDA) algorithm.
The VDA, further explained in Kräußling *et al.* [2004], starts with a set of validated measurements $z_i(k-1)$, with $i = 1, \ldots, m_k$. Then, when a new set of measurements $z_i(k)$, comes available, the lengths of the existing tracks from the starting point to the new measurement $z_i(k)$, through the old measurement $z_i(k-1)$ is calculated. As a result, for each measurement $z_i(k)$, the length of the shortest path to this measurement is obtained. The difference with the NNSF lies in the fact that the NNSF only calculates one path each recursion step.

Alternative implementations can be found in, *e.g.*, Gad & Farooq [2003], where two different implementations of the VDA algorithm are compared. These algorithm use the same VA, however, the transition from the data association problem to a solution using the VA is different. They use joint likelihood probabilities as 'distances' between measurements. The minimizing step now results in an optimal joint likelihood probability instead of the length of the shortest track.

3.8 Data clustering

If multiple sensors are used to measure the same variable, *e.g.*, the position of a robot, the measurements typically vary, *e.g*, due to measurement noise or localization errors. To assist the data association algorithm, a class of algorithms that is regularly used is data clustering, already briefly mentioned in Section 3.6 about the MHT. A data clustering algorithm groups measurements in clusters in such a way that each cluster only contains measurements originating from one target, thereby significantly reducing the data. From thereon, the data association algorithm continues as explained earlier. Data clustering algorithms are not considered throughout this literature study. Details about data clustering can be found in, *e.g.*, Gan *et al.* [2007].

Chapter 4

Tracking

4.1 Bayesian tracking approach

In this section, the state space model introduced in Chapter 2 is used as a starting point. In the Bayesian approach, the states are assumed to be random variables with a posterior pdf $p(x_k|z_{1:k})$, where $z_{1:k} \triangleq \{z_i,\ i = 1,\ \ldots, k\}$. More general, for any variable $(\cdot)$, $(\cdot)_{1:k} \triangleq \{(\cdot)_i,\ i = 1,\ \ldots, k\}$ The pdf $p(x_k|z_{1:k-1})$ is approximated using (1) a prediction and (2) an update step.

1. In the prediction step, the Chapman-Kolmogorov equation will be used to calculate the prior pdf $p(x_k|z_{1:k-1})$, *i.e.*, the predicted pdf of x_k on the basis of measurements up to time $k-1$. If $p(x_{k-1}|z_{1:k-1})$ is available and the system in Equation (2.1) is assumed to be Markov, then the Chapman-Kolmogorov equation is:

$$p(x_k|z_{1:k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|z_{1:k-1})dx_{k-1} \tag{4.1}$$

2. At time step k, a new measurement z_k becomes available and Bayes' rule can be used to update the prior pdf to the posteriori probability $p(x_k|z_{1:k})$, using the conditional probability $p(x_k|z_{1:k-1})$:

$$p(x_k|z_{1:k}) = \frac{p(z_k|x_k)p(x_k|z_{1:k-1})}{p(z_k|z_{1:k-1})}, \tag{4.2}$$

   where the denominator is a normalizing factor:

$$p(z_k|z_{1:k-1}) = \int p(z_k|x_k)p(x_k|z_{1:k-1})dx_k. \tag{4.3}$$

   A more detailed explanation of Bayes' rule is given in Appendix A.2.

If this problem of recursively calculating the posterior pdf is solved exactly, the optimal Bayesian solution is obtained. Unfortunately, this optimal solution only exists in a restricted set of cases since it involves the evaluation of complex high-dimensional integrals. If the problem does not lie within such a restricted set of cases, suboptimal solutions that can be used.

4.2 Linear Kalman filter

The Linear Kalman filter (LKF) gives the optimal Bayesian solution to the state estimation problem, if the posterior pdf at every time step is Gaussian. Furthermore, f_k and h_k in Equations (2.1) and (2.2) should be known and linear in x_{k-1} and v_{k-1}, respectively x_k (and in the control input, if present). The noise vectors w_k, and v_{k-1} and w_k should be drawn from zero mean Gaussian distributions with known covariances Q_k and R_k. It is widely used, see *e.g.*, Farrell & Ioannou [2001] or Yu *et al.* [2005], explained in Appendix C.1 and already mentioned in the previous chapter. The main advantages are then optimal Bayesian solution and the low computational complexity and memory requirements. An obvious drawback is the above mentioned assumptions.

4.3 Extended Kalman filter

In many practical situations, Equations (2.1) and (2.2) are nonlinear and for that reason, the LKF as introduced in Section 4.2 cannot be used directly. Alternatives using the same Bayesian framework are called sub-optimal Bayesian methods. One obvious sub-optimal Bayesian method is using a local linear approximation at each time step and then applying the linear Kalman filter. This is the basic idea of the extended Kalman filter (EKF), used in, among others, Bianchi & Tinnirello [2003]. This means that

$$p(x_{k-1}|z_{1:k-1}) \approx \mathcal{N}(x_{k-1};\, m_{k-1|k-1},\, P_{k-1|k-1}) \tag{4.4}$$

$$p(x_k|z_{1:k-1}) \approx \mathcal{N}(x_k;\, m_{k|k-1},\, P_{k|k-1}) \tag{4.5}$$

$$p(x_k|z_{1:k}) \approx \mathcal{N}(x_k;\, m_{k|k},\, P_{k|k}), \tag{4.6}$$

where $\mathcal{N}(x;\, m,\, P)$ is a Gaussian density with argument x, mean m, and covariance P. The matrices F_k and H_k in the standard LKF are approximated by the (local) linearizations $\hat{F}_k$ and $\hat{H}_k$ around the means $m_{k-1|k-1}$, respectively $m_{k|k-1}$. Note that this approximation corresponds to a first order Taylor expansion. The increasing complexity has prohibited the widespread use of higher order Taylor expansions. If the system under consideration has weak nonlinearities only, this suboptimal algorithm can be very effective.
There are a few major drawbacks using this approach. If the nonlinearities become severe, the performance of the filter can decrease rapidly. This might eventually lead to diverging estimates. It is not easy to determine this beforehand. Furthermore, an obvious requirement is the existence of a linear approximation, *i.e.*, the absence of discontinuities or singularities. More details about the disadvantages of an EKF can be found in Julier & Uhlmann [2004].

4.4 Unscented Kalman filter

A second sub-optimal Bayesian method that is based on the LKF is the unscented Kalman Filter (UKF), that falls within the group of Sigma-Point Kalman Filters. According to Julier & Uhlmann [2004], the basic idea of this filter is that "it is easier to approximate a probability distribution than it is to approximate an arbitrary nonlinear function or transformation". While considering the spread of a random variable the UKF tends to be more accurate than the first order Taylor series linearization used in the EKF, according to van der Merwe [2004]. It is used in, *e.g.*, Wan & van der Merwe [2000]. A brief overview of the algorithm is as follows:

1. Select a minimum number of L points, called sigma points, where $L = 2n_x + 1$ and n_x again is the state dimension. Since this selection is made deterministically, the sigma points can be chosen from a Gaussian distribution with a desired mean or covariance, which limits the required number of sigma points.
2. The nonlinear equation is used to transform the sigma points, leading to a set of transformed points.
3. The transformed points are used to re-approximate the (nonlinearly transformed) mean and covariance of the Gaussian distribution.

Contrary to the EKF, the UKF can deal with severe nonlinearities while its computational complexity has the same order of magnitude. Disadvantage is that again a Gaussian distribution is assumed. If the true density is non-Gaussian, it is very likely that neither the UKF nor the LKF and the EKF are able to describe it well.
All Kalman filters that are explained up till this point approximate state vectors with a fixed length only, *i.e.*, they can not be used to track a varying number of targets simultaneously. An obvious solution is using a separate filter for each target. Another algorithm, usually the data association algorithm, has to be used in order to determine the number of targets. This is done by, *e.g.*, Hou *et al.* [2009] and Straw *et al.* [2010].

4.5 Grid-based methods

Another way to find the optimal pdf $p(x_k|z_{1:k})$, using the Bayesian framework is using a grid-based method, as explained in, *e.g.*, Zheng & Xue [2009]. This method assumes a discrete state space with a finite number of states. The posterior pdf is written as

$$p(x_{k-1}|z_{1:k-1}) = \sum_{i=1}^{N_s} w^i_{k-1|k-1}\delta(x_{k-1} - x^i_{k-1}), \tag{4.7}$$

where N_s is the number of states, $\delta(\cdot)$ is the Dirac delta measure and for each state x^i_{k-1}, $w^i_{k-1|k-1}$ is the conditional probability, *i.e.*, the probability that $x_{k-1} = x^i_{k-1}$ given measurements $z_{1:k-1}$. Substitution of this equation into the prediction Equation (4.1) and the update Equation (4.2), leads to the grid-based prediction and update equation.

4.6 Approximate grid-based methods

In approximate grid-based methods, a continuous state space is decomposed in N_s cells, each with its own center. The center of the jth cell at time index k is given by $\bar{x}^j_k$. The difference with Section 4.5 is that now the calculation of the probabilities $w^i_{k|k}$ involves the center of the cell $\bar{x}^j_k$ instead of the state x^j_k. As a result, Equation (4.7) changes to

$$p(x_{k-1}|z_{1:k-1}) \approx \sum_{i=1}^{N_s} w^i_{k-1|k-1}\delta(x_{k-1} - x^i_{k-1}), \tag{4.8}$$

One of the disadvantages of this method is that the computational cost increases rapidly with the dimensionality of the state space, since the grid should be sufficiently dense to represent the continuous state space. Furthermore, the state space should be defined in

advance and the grid should have a high constant resolution over the whole domain, or prior knowledge about regions with a high probability has to be used. Probably due to these reasons, grid-based methods are hardly used in recent literature.

4.7 Sequential Monte Carlo methods

A more general way to solve the tracking problem is using sequential Monte Carlo (SMC) methods, already mentioned in the previous chapter. SMC are very popular, *e.g.*, Gustafsson *et al.* [2002], Hue *et al.* [2002], and Thrun [2002], and used to do approximations in many fields of research. Their main advantage is their ability to approximate complex high dimensional densities, *i.e.*, they can be used to approximate states of non-linear dynamical models and non-Gaussian noise. Several closely related SMC algorithms are widely used under different names, *e.g.*, bootstrap filter, condensation, particle filters, and survival of the fittest. A detailed explanation can be found in Appendix C.2.

A well-known technique that can be incorporated into the particle filter is Rao-Blackwellization, explained in *e.g.*, Doucet *et al.* [2001]. The idea is to reduce the size of the state space by marginalizing out some of the variables analytically. Analytical updates are done using the (efficient) LKF and as a result the remaining space that has to be sampled from is smaller leading to a much lower required number of samples. An extension of the Rao-Blackwellized particle filter to tracking an unknown varying number of targets is introduced in Särkkä *et al.* [2007]. A continuation that dropped the independence assumption between data associations is introduced in von Hoyningen-Huene & Beetz [2009], where a real-time implementation is realized. von Hoyningen-Huene & Beetz [2009] combine the advanced particle filter with LKFs, which means that at some point, the Kalman assumptions are made. This might be done to keep the computational costs limited. It could be interesting to see to what extend these LKFs are required to keep real-time implementations possible.

4.8 Maximum likelihood and maximum a posteriori

Bayesian probability is just one way of interpreting probability theory. Two important non-Bayesian methods are maximum likelihood (ML) and maximum a posteriori (MAP). ML tries to find the vector x that maximizes the likelihood function. In other words, given an observation vector y, select the values for x that maximize the probability of observing y. This involves varying x and calculating the probability of the observed data for that x.

MAP calculates the most probable value of x given the data y. Contrary to Bayesian methods where densities are involved, ML and MAP result in point estimates. As explained in Bishop [2006], ML corresponds to minimizing the sum of squares error function between measurements y and predicted values of a polynomial approximation whose coefficients are included in x. MAP corresponds to minimizing the regularized sum of squares error, *i.e.*, a sum of squares error with an additional term that penalties large coefficients x (the order of x). Recent literature using MAP is Saha *et al.* [2009], where the MAP is estimated using a particle based filter and Ma *et al.* [2009], where a MAP formulation is combined with a sort of track splitting algorithm.

4.9 Probability hypothesis density filter

An approach that is specifically designed for multi-target tracking is a recursive Bayesian method using the probability hypothesis density (PHD), explained in Mahler [2003]. This method is an extension of the Bayesian analysis and employs Finite Set Statistics (FISST). The goal Mahler [2003] is to find a multi-target tracking strategy analogous to the LKF in single-target tracking. This strategy is found using the PHD and, furthermore, it is shown that the PHD is a best-fit approximation in an information-theoretic sense. An explanation of the strategy can be found in Appendix C.3.
An advantage of this method is that it is a multi-sensor multi-target method that includes sensor models and the appearance and disappearance of targets. Therefore, it automatically solves the data association problem. Furthermore, in his conclusion Mahler [2003] states that, based on his own work, "it is anticipated that implemented PHD filters will be most useful in high-density applications where multi hypothesis trackers begin to fail, *e.g.*, cluster tracking and group-target tracking". Among the assumptions are a good signal-to-noise ratio (SNR) and independent target movements.
A paper implementing a PHD based method as described by Mahler [2003] using particle filters is Sidenbladh [2003], where only near-real-time performance was obtained. An improved implementation was introduced in Lin *et al.* [2006], where, among other things, track labeling was added, resulting in an improved performance and track continuity. The only test was an one dimensional multi-target tracking simulation and it is unclear if real-time performance can be obtained for higher dimensional problems. However, during this simulation, the PHD filter with labeling outperformed the "traditional MHT/assignment approach". Unfortunately, no details about this traditional approach are given, *i.e.*, it is not known if it outperforms state of the art MHT filters. A continuation that incorporates fusing the information of multiple particle filters is Danu *et al.* [2008], while recommendations regarding the costs of such implementations are given in Danu *et al.* [2009]. Whiteley *et al.* [2007] state that they introduce a more efficient particle filter implementation of the PHD filter, relative to particle filter implementations like suggested in Sidenbladh [2003], and supports this statement with a numerical simulation.
At the same time, a cardinalized PHD (CPHD) recursion was developed by Mahler [2007a]. The CPHD recursion, like the PHD filter, propagates the first-order statistics in the states of targets, but also the entire probability distribution on the number of targets (called cardinality distribution). In a dissertation by Vo [2008], a SMC implementation was introduced that allows dropping the linearity assumption. In this dissertation, during a simulation with a non-linear dynamical model the CPHD filter outperforms the PHD filter. However, it is not clear if a real-time implementation is possible in more complex scenarios, since the number of targets was limited and the process noise was modeled by using a Gaussian distribution. A track continuity extension as in Lin *et al.* [2006], but also in Clark & Bell [2007], is a desirable extension to CPHD recursion.
In Svensson *et al.* [2009], a Gaussian mixture CPHD implementation was compared with a MHT in a target tracking scenario. Unfortunately, the SMC CPHD recursion was not implemented. Due to the cardinality distribution the CPHD recursion responded faster to changes in the number of targets. The slow response of the MHT on the other hand had a lower standard deviation in the number of objects. Furthermore, the MHT resulted in a more stable output, *i.e.*, it was able to keep track over short periods of target invisibility. Both algorithms enabled real-time implementation and the overall performance was said to

be equally well. More details can be found in Svensson *et al.* [2009].
If the dynamic and measurement models are highly nonlinear, better results are obtained by propagating the multi-target posterior density based on parameterized density approximations via multi-Bernouilli random finite sets. Again, the method was initiated by Mahler, see Mahler [2007b], and improved by Vo, see Vo [2008] or Vo *et al.* [2009].
However, this parameterized density approximation based approach only works with a reasonably high signal-to-noise ratio (SNR). Again, an extension to algorithms that perform track estimation is desired.

Chapter 5

Conclusions and Recommendations

In the previous chapters, it was tried to give a complete overview of the state of the art in multi-sensor multi-target data association and target tracking. All sections include references in which the methods are explained in more detail. Furthermore, the strong points of each method are given together with the drawbacks. This chapter briefly repeats the methods that were found most promising. After that some recommendations are done.

5.1 Conclusions

- The linear Kalman filter (LKF), the extended Kalman filter (EKF), and the unscented Kalman filter (UKF) are widely used, due to their low computational costs and the good results in many practical situations. However, if severe nonlinear dynamics and non-Gaussian noise are involved, particle filters are preferred, since these do not have the restricting assumptions the Kalman filters have. Particle filters have high computational costs and for that reason the number of particle filters should be kept as low as possible.
- The multi hypothesis filter (MHF) is considered to be a successful data association method. To allow a real-time implementation of the MHF, the size of the hypothesis tree should be limited, *e.g.*, by clustering or hypothesis pruning. Combining this method with Kalman filters results in a robust method that is able to deal with short periods of target invisibility and allows a real-time implementation.
- A successful alternative is the joint probabilistic data association filter (JPDAF). This filter has a low computational complexity, relative to the MHT and the sample-based JPDAF (SJPDAF) allows dropping the linearity assumptions and tracking an unknown number of targets, by using particle filters. The generalization to multiple sensors and arbitrary proposal distributions in Monte Carlo JPDAF (MC-JPDAF) is beneficial, however, it assumes a fixed and known number of targets which is a major drawback.
- The probability hypothesis density (PHD) filter is a relatively new filter that has already proven its value in some applications. The particle filter implementation with track labeling is considered to be an improved version of the standard PHD filter, however, it is not entirely clear if real-time performance can be obtained in complex situations, *i.e.*, high and varying number of targets, unstructured environment, and noisy measurements. The cardinality PHD (CPHD) filter, a PHD filter with an

additional distribution for the number of targets, and its state of the art extensions is an interesting filter that is developed recently. A track labeling extension is desired but not yet available. A comparison between a MHF and one particular CPHD filter, performed by Svensson *et al.* [2009], did not result in a obvious winner.

5.2 Recommendations

These conclusions lead to the following recommendations.

- Many algorithms are introduced during many years of research. It is nearly impossible to rank the performance of the different methods based on theory only. It would be very useful to compare the different methods and see which method is best and how exactly the performance of the methods differs in various scenarios.
- It can be useful to extend existing methods, as indicated throughout this report, thereby allowing dropping some of the restricting assumptions. How the state of the art can be extended while maintaining real-time performance it not clear yet. The recommended comparison will point out if the various methods should be extended and in which directions.

References

Abolmaesumi, P., & Sirouspour, M.R. 2004. An Interacting Multiple Model Probabilistic Data Association Filter for Cavity Boundary Extraction From Ultrasound Images. *IEEE Transactions on Medical Imaging*, **23**(6), 772–784.

Anderson, Eric C. 1999. *Monte Carlo Methods and Importance Sampling.* University of California, Berkeley. Lecture Notes for Stat 578C.

Arulampalam, M. Sanjeev, Maskell, Simon, Gordon, Neil, & Clapp, Tim. 2002. A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking. *IEEE Transactions on Signal Processing*, **50**(2), 174–188.

Bar-Shalom, Yaakov, & Fortmann, Thomas E. 1988. *Tracking and Data Association.* Mathematics in Science and Engineering, vol. 179. Academic Press.

Bianchi, Giuseppe, & Tinnirello, Ilenia. 2003. Kalman Filter Estimation of the Number of Competing Terminals in an IEEE 802.11 network. *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE*, **2**, 844–852.

Bishop, Christopher M. 2006. *Pattern Recognition and Machine Learning.* Information Science and Statistics. Springer.

Blackman, Samuel S. 2004. Multiple Hypothesis Tracking For Multiple Target Tracking. *IEEE Aerospace and Electronic Systems Magazine*, **19**(1), 5–18.

Calvetti, Daniela, & Somersalo, Erikki. 2007. *Introduction to Bayesian Scientific Computing, Ten Lectures on Subjective Computing.* Surveys and Tutorials in the Applied Mathematical Sciences, vol. 2. Springer.

Clark, Daniel E., & Bell, Judith. 2007. Multi-Target State Estimation and Track Continuity for the Particle PHD Filter. *IEEE Transactions on Aerospace and Electronic Systems*, **43**(4), 1441–1453.

Cox, Ingemar J., & Hingorani, Sunita L. 1996. An Efficient Implementation of Reid's Multiple Hypothesis Tracking Algorithm and Its Evaluation for the Purpose of Visual Tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **18**(2), 138–150.

Danu, D., Kirubarajan, T., Lang, T., & McDonald, M. 2008. Multisensor Particle Filter Cloud Fusion for Multitarget Tracking. *Pages 1–8 of: 11th International Conference on Information Fusion.*

Danu, D., Kirubarajan, T., & Lang, T. 2009. Wasserstein Distance for the Fusion of Multisensor Multitarget Particle Filter Clouds. *Pages 25–32 of: 12th International Conference on Information Fusion.*

Doucet, Arnaud, Godsill, Simon, & Andrieu, Christophe. 2000. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and computing*, **10**(3), 197–208.

Doucet, Arnaud, Freitas, Nando De, & Gordon, Neil. 2001. *Sequential Monte Carlo Methods in Practice.* Springer.

Farrell, Brian F., & Ioannou, Petros J. 2001. State Estimation Using a Reduced-Order Kalman Filter. *Journal of the Atmospheric Sciences*, **58**, 3666–3680.

Gad, Ahmed, & Farooq, M. 2003. Single Target Tracking in CLutter: Performance Comparison between PDA and VDA. *Pages 1266–1273 of: Proceedings of the Sixthe International Conference of Information Fusion*, vol. 2.

Gan, Guojun, Ma, Chaoqun, & Wu, Jianhong. 2007. *Data Clustering, Theory, Algorithms, and Applications.* Society for Industrial and Applied Mathematics.

Gordon, N.J., Salmond, D.J., & Smith, A.F.M. 1993. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proceedings-F*, **140**(2), 107–113.

Gorji, Aliakbar, Menhaj, Mohammad Bagher, & Shiry, Saeed. 2007. Multiple Target Tracking for Mobile Robots Using the JPDAF Algorithm. *19th IEEE International Conference on Tools with Artificial Intelligence*, **1**, 137–145.

Gustafsson, Fredrik, Gunnarsson, Fredrik, Bergman, Niclas, Forssell, Urban, Jansson, Jonas, Karlsson, Rickard, & Nordlund, Per-Johan. 2002. Particle Filters for Positioning, Navigation, and Tracking. *IEEE Transactions on Signal Processing*, **50**(2), 425–437.

Hempel, Christian G., & Pacheco, Jason. 2009. Performance analysis of the Probabilistic Multi-hypothesis Tracking algorithm on the SEABAR data sets. *Pages 1830–1836 of: 12th International Conference on Information Fusion.*

Hou, Sheng-Yun, Hung, Hsien-Sen, Chang, Yuan-Chang, & Chang, Shun-Hsyung. 2009. Multitarget Tracking Algorithms Using Angle Innovations and Extended Kalman Filter. *WTOS*, **8**(3), 420–429.

Hue, C., Cadre, J-P. Le, & Pérez, P. 2002. Tracking Multiple Objects with Particle Filtering. *IEEE Transactions on Aerospace and Electronic Systems*, **38**(3), 791–812.

Julier, Simon J., & Uhlmann, Jeffrey K. 2004. Unscented Filtering and Nonlinear Estimation. *Proceedings of the IEEE*, **92**(3), 401–422.

Kalman, R.E. 1960. A New Aproach to Linear Filtering and Prediction Problems. *IEEE Transactions on Signal Processing*, **82**(Series D), 35–45.

Kirubarajan, Thiagalingam, & Bar-Shalom, Yaakov. 2004. Probabilistic Data Association Techniques for Target Tracking in CLutter. *Proceedings of the IEEE*, **92**(3), 536–557.

Kräußling, Andreas, Schneider, Frank E., & Wildermuth, Dennis. 2004. Tracking Expanded Objects Using the Viterbi Algorithm. *Pages 566–571 of: Proceedings of the Second IEEE International Conference on Intelligent Systems*, vol. 2.

Leung, Henry, Hu, Zhijian, & Blanchette, Martin. 1999. Evaluation of Multiple Radar Target Trackers in Stressful Environments. *IEEE Transactions on Aerospace and Electronic Systems*, **35**(2), 663–674.

Lin, L., Bar-Shalom, Y., & Kirubarajan, T. 2006. Track Labeling and PHD Filter for Multitarget Tracking. *IEEE Transactions on Aerospace and Electronic Systems*, **42**(3), 778–795.

Ma, Yunqian, Yu, Qian, & Cohen, Isaac. 2009. Target Tracking with Incomplete Detection. *Computer Vision and Image Understanding*, **113**(5), 580–587.

Mahler, Ronald. 2007a. PHD Filters of Higher Order in Target Number. *IEEE Transactions on Aerospace and Electronic Systems*, **43**(4), 1523–1543.

Mahler, Ronald. 2007b. *Statistical Multisource-Multitarget Information Fusion.* Artech House.

Mahler, Ronald P.S. 2003. Multitarget Bayes Filtering via First-Order Multitarget Moments. *IEEE Transactions on Aerospace and Electronic Systems*, **39**(4), 1152–1178.

Rabiner, Lawrence R. 1989. A Tutorial on Hidden Markov Models ans Selected Application in Speech Recognition. *Proceedings of the IEEE*, **77**(2), 257–286.

Reid, Donald B. 1979. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, **24**(6), 843–854.

Saha, S., Boers, Y., Driessen, H., Mandal, P.K., & Bagchi, A. 2009. Particle Based MAP State Estimation: A Comparison. *Pages 278–283 of: 12th International Conference on Information Fusion.*

Särkkä, Simo, Vehtari, Aki, & Lampinen, Jouko. 2007. Rao-Blackwellized Particle Filter for Multiple Target Tracking. *Special Issue on the Seventh International Conference on Information Fusion-Part II*, **8**(1), 2–15.

Schultz, Dirk, Burgard, Wolfram, Fox, Dieter, & Cremers, Armin B. 2003. People Tracking with Mobile Robots Using Sample-based Joint Probabilistic Data Association Filters. *The International Journal of Robotics Research*, **22**(2), 99–116.

Sidenbladh, Hedvig. 2003. Multi-Target Particle Filtering for the Probability Hypothesis Density. *Sixth International Conference on Information Fusion*, 800–806.

Smith, P., & Buechler, G. 1975. A Branching Algorithm for Discriminating and Tracking Multiple Objects. *IEEE Transactions on Automatic Control*, **AC-20**, 101–104.

Song, Taek Lyul, Lee, Dong Gwan, & Ryu, Jonha. 2005. A probabilistic nearest neighbor filter algorithm for tracking in a clutter environment. *Signal Processing*, **85**(10), 2044–2053.

Straw, Andrew D., Branson, Kristin, Neumann, Titus R., & Dickinson, Michael H. 2010. Multi-camera Realtime 3D Tracking of Multiple Flying Animals. *CoRR*, **abs/1001.4297**. Submitted, preprint.

Streit, R.L. 2006. The PMHT and Related Applications of Mixture Densities. *In: Proceedings of the International Conference on Information Fusion.*

Svensson, Daniel, Wintenby, Johannes, & Svensson, Lennart. 2009. Performance Evaluation on MHT and GM-CPHD in a Ground Target Tracking Scenario. *Pages 300–307 of: 12th International Conference on Information Fusion.*

Thrun, Sebastian. 2002. Particle Filters in Robotics. *Proceesings of Uncertainty in AI (UAI).*

van der Merwe, Rudolph. 2004. *Sigma-Point Kalman Filters for Probabilistic Inference in Dynamic State-Space Models.* Ph.D. thesis, Oregon Health & Science University.

Vermaak, Jaco, Godsill, Simon J., & Pérez, Patrick. 2005. Monte Carlo Filtering for Multi-Target Tracking and Data Association. *IEEE Transactions on Aerospace and Electronic Systems*, **41**(1), 309–332.

Viterbi, Andrew J. 1967. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, **13**(2), 260–269.

Vo, Ba Tuong. 2008. *Random Finite Sets in Multi-Object Filtering.* Ph.D. thesis, School of Electrical, Electronic and Computer Engineering, The University of Western Australia.

Vo, Ba-Tuong, Vo, Ba-Ngu, & Cantoni, Antonio. 2009. The Cardinality Balanced Multi-Target Multi-Bernouilli Filter and Its Implementations. *IEEE Transactions on Signal Processing*, **57**(2), 409–423.

von Hoyningen-Huene, Nicolai, & Beetz, Michael. 2009. Rao-Blackwellized Resampling Particle Filter for Real-Time Player Tracking in Sports. *Pages 464–471 of: VISAPP, INSTICC.*

Wan, Eric A., & van der Merwe, Rudolph. 2000. The Unscented Kalman Filter for Nonlinear Estimation. *The IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC*, 153–158.

Whiteley, Nick, Singh, Sumeetpal, & Godsill, Simon. 2007. Auxiliary Particle Implementation of the Probability Hypothesis Density Filter. *Pages 510–515 of: Proceedings of the 5th International Symposium on image and Signal Processing and Analysis.*

Willett, Peter, Ruan, Yanhua, & Streit, Roy. 2002. The PMHT and Related Applications of Mixture Densities. *IEEE Transactions on Aerospace and Electronic Systems*, **38**(3), 738–754.

Yu, Kent K.C., Watson, N.R., & Arrillaga, J. 2005. An Adaptive Kalman Filter for Dynamic Harmonic State Estimation and Harmonic Injecion Tracking. *IEEE Transactions on Power Delivery*, **20**(2), 1577–1584.

Zhang, Y., Leung, H., Lo, T., & Litva, J. 1996. Distributed sequential nearest neighbour multitarget tracking algorithm. *IEE Proceedings on Radar, Sonar and Navigation*, **143**(2), 255–260.

Zheng, Nanning, & Xue, Jianra. 2009. *Statisical Learning and Pattern Analysis for Image and Video Processing*. 1 edn. Advances in Pattern Recognition. Springer London.

Appendix A

Probability Theory

A.1 Probability theory

This section briefly introduces some relevant elements of the probability theory that is referred to in this report. It is based on Calvetti & Somersalo [2007], so for more details is referred to this reference. Starting point is the real valued random variable X. If the probability space is denoted Ψ, with elements ψ, then $x = X(\psi)$, $\psi \in \Psi$ is called a realization of X. Now let E be a subset of Ψ, *i.e.*, $E \subset \Psi$, then the probability $P(E)$, with $0 \leq P(E) \leq 1$, denotes the probability that X takes a value in E. If $A \cap B = \emptyset$ and $A,\ B \subset \Psi$, then

$$P(A \cup B) = P(A) + P(B), \tag{A.1}$$

and the events A and B are called independent if

$$P(A \cap B) = P(A)P(B). \tag{A.2}$$

The probability distribution of X, denoted $\mu_X(B)$ for each $B \subset \mathbb{R}$, is the probability of event $x \in B$ and it measures the size of the subspace of Ψ mapped onto B by the random variable X. It is linked to the probability density function (pdf) $p_X(x)$ as follows

$$\mu_X(B) = \int_B p_X(x)dx. \tag{A.3}$$

The subscript X will be omitted, unless it is found necessary to specify the dependency, so $p_X(x) = p(x)$. The discrete time equivalent of the pdf will be called probability mass function (pmf) and is denoted:

$$p(x) = \sum_{i=1}^{n} \mu_i \delta(x - \xi), \tag{A.4}$$

where the point mass is defined $\mu_i \triangleq P(X = \xi_i)$ and $\delta(x)$ is the Dirac delta function. The discrete time random variable X can take values from the set $\{\xi_i, i = 1, \ldots, n\}$ and the sum of all point masses μ_i equals one

A.2 Definitions

With the theory from Section A.1, some additional definitions can be specified:

Conditional probability The probability that A happens provided that B happens:

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \tag{A.5}$$

Conditional probability density The probability of X assuming that $Y = y$, given X and Y:

$$p(x|y) = \frac{p(x,y)}{p(y)}, \quad p(y) \neq 0. \tag{A.6}$$

Joint probability density The probability that $x \in A$ and at the same time $y \in B$, given X and Y:

$$P(X \in A, Y \in B) = \int \int_{A \times B} p(x,y) dx dy \tag{A.7}$$

Marginal density The marginal density of X is the pdf of X regardless of Y, given X, Y, and p(x,y):

$$p(x) = \int_{\mathbb{R}} p(x,y) dy. \tag{A.8}$$

The marginal density of Y is defined analogously.

First order moment Also called expectation or weighted average of X, is the center of mass of the pdf:

$$E(x) = \int_{\mathbb{R}^n} x p(x) dx, \tag{A.9}$$

where n is the length of vector X.

Second order moment Also called variance of X:

$$\sigma_X^2 = E\{(X - E[X])^2\}, \tag{A.10}$$

in the multivariate case, this changes to the (symmetric, positive semi-definite) covariance matrix, which expresses the to extend to which the elements of x vary together:

$$\text{cov}(X) = \int_{\mathbb{R}^n} (x - E[x])(x - E[x])^T p(x) dx. \tag{A.11}$$

Using Equation (A.6), the derivation of Bayes' rule is straightforward:

$$p(x,y) = p(x|y)p(y) = p(y|x)p(x) \quad \Rightarrow \quad p(x|y) = \frac{p(y|x)p(x)}{p(y)}. \tag{A.12}$$

To fully understand Bayes' rule, a brief explanation is given for the general case of approximating parameters collected in a vector x that are part of a model that tries to predict the outputs of some process, from which (noisy) measurements y are taken at each time step. In Bayes' rule, the probability $p(x|y)$ is called the posterior probability. This can be explained as the probability of the parameters in x given measurements y. The probability $p(y|x)$ is the probability of the measurements given the parameters in x and is called likelihood function. In other words, it expresses how probable the observed data is given the parameters x. The probability $p(x)$ is called prior probability distribution. The denominator is the probability $p(y)$ of the measurements independent of the parameters x, this normalizes the posterior distribution. This equation can be used to up-date a probability $p(x)$ each time new measurements become available, it takes both old and new observations into account.

A.3 Neural networks

In a neural network, first the activations a_j are constructed. These are linear combinations of the input variables x, where each input variable is weighted by its its own parameter, called weight and denoted by $w_{ji}^{(1)}$, $i \geq 1$. If the system has D inputs x_i, then the M activations are calculated have the form

$$a_j = \sum_{i=0}^{D} w_{ji}^{(1)} x_i. \tag{A.13}$$

Note that the input x_0 does not exists, this value is introduced at this point and given a value $x_0 = 1$. Its weighting parameter is called bias. Furthermore, the number of activations M can be selected by the user. The superscript (1) in this equation indicates that the parameters are in the first layer of the network.
The next step is a nonlinear transformation. The activations are transformed using an activation function $h(\cdot)$ resulting in what is called the hidden units: $z_j = h(a_j)$. Usually, the transformation is performed by means of a logistic sigmoid $\sigma(a)$, defined as

$$\sigma(a) = \frac{1}{1 + \exp(-a)}, \tag{A.14}$$

or a hyperbolic tangent $\tanh(a)$, defined as

$$\tanh(a) = \frac{\exp(2a) - 1}{\exp(2a) + 1}. \tag{A.15}$$

Then, in the third step, these hidden units z_j are linearly combined resulting in K output unit activations. The number K generally is determined by the dimensionality of the data set. This step is similar to Equation (A.13), however, now superscripts (2) are used to indicate the second layer of the network.

$$a_k = \sum_{j=0}^{M} w_{kj}^{(2)} z_j. \tag{A.16}$$

In the fourth and final step, the K output unit activations are transformed to the network outputs y_k using the logistic sigmoid function σ. If all weights and biases are collected in one vector w, the overall network function, including all four steps, becomes:

$$y_k(x, w) = \sigma \left(\sum_{j=0}^{M} w_{kj}^{(2)} h \left(\sum_{i=0}^{D} w_{ji}^{(1)} x_i \right) \right). \tag{A.17}$$

A visual representation of this neural network is given in Figure A.1. This neural network is also known as multilayer perceptron (MLP). A new layer can be added by introducing an additional linear combination and nonlinear transformation.
A very important decision when using a neural network is the selection of the parameters in w. Usually, these nonconstant parameters are calculated by minimizing a cost function that typically has many local minima. Various ways to tackle this problem exist, *e.g.*, the maximum likelihood, maximum a posteriori, and the Bayesian approaches, all explained in Bishop [2006].

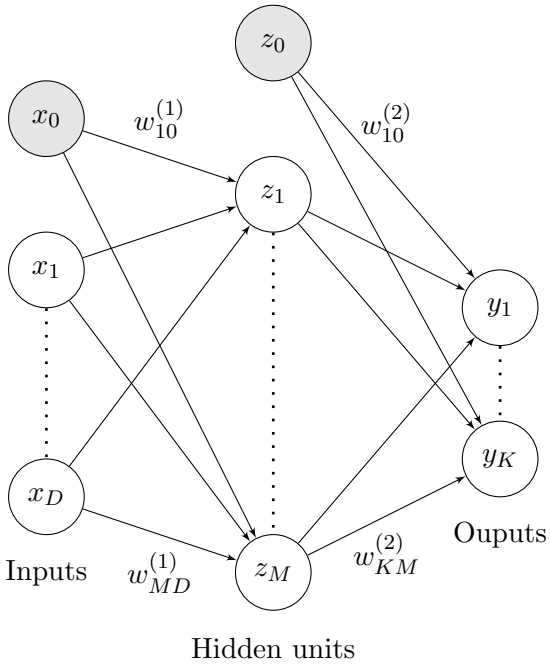

Figure A.1: Neural network diagram.

Appendix B

Details Data Association Related Methods

B.1 Probabilistic data association filter

The starting point of the PDAF is a set of m_k validated measurements, obtained by using a gate as described in Section 3.1. Now the event θ_0 means that none of the m_k validated measurements is target-originated, whereas $\theta_i(k)$, with $i = 1, \ldots, m_k$, means that validated measurement $z_i(k)$ is target-originated. The probabilities corresponding to these events are represented by $\beta_i(k)$, with $i = 0, \ldots, m_k$, such that the sum

$$\sum_{i=0}^{m_k} \beta_i(k) = 1. \tag{B.1}$$

A linear Kalman filter (LKF) algorithm is applied to each of the validated measurements. This means that for each validated measurement $z_i(k)$, the innovation $\nu_i(k)$, the covariance matrix of this innovation $S_i(k)$, and a Kalman gain $K_i(k)$ calculated using this covariance matrix, are calculated. A detailed explanation of the LKF and these calculations is given in Appendix C.1. Now, the state conditioned on the event θ_i, *i.e.*, measurement z_i is target-originated, is given by

$$\hat{x}_{k|k} = \sum_{i=0}^{m_k} \beta_i(k)\hat{x}_i(k), \tag{B.2}$$

and the state update equation, Equation (C.11) in the LKF, is replaced by

$$x_{k|k} = \hat{x}_{k|k-1} + \sum_{i=1}^{m_k} \beta_i(k)K_i(k)\nu_i(k). \tag{B.3}$$

Note that in Equation (B.2), both the probability β_i and the state x_i depend on the innovation, making this equation (highly) nonlinear. For details about intermediate steps is referred to Bar-Shalom & Fortmann [1988]. Next step is the calculation of the probabilities β. Again, details are omitted but some explanation about the steps that are required is given.
The number of validated measurements and the locations of the these measurements give probabilistic information. This information is used to write the expressions for $\beta_i(k)$ using

Bayes' law. Then, with the assumption that the incorrect measurements have a uniform pdf, this expression is rewritten. The newly obtained expression depends on the following variables

- The gate probability P_G.
- The size of the validation region $\tilde{V}_k$.
- The number of validated measurements m_k.
- A zero-mean normal distribution with argument $\nu_i(k)$ and covariance S_k.
- The probability γ_i of event θ_i conditioned on only the total number of validated measurements m_k.

The expression for γ_i includes the pmf of the number of false measurements. For this pmf, a parametric or a nonparametric model can be used. The choice that is made at this point influences b in the final expressions

$$\beta_i = \frac{e_i}{b + \sum_{j=1}^{m_k} e_j}, \quad i = 1, \ldots, m_k \tag{B.4}$$

$$\beta_0 = \frac{b}{b + \sum_{j=1}^{m_k} e_j}. \tag{B.5}$$

The variables e and b are introduced for ease of writing and the expressions can be found in Bar-Shalom & Fortmann [1988, Equations (6-44)-(6-46)].

B.2 Joint probabilistic data association filter

First, the event θ_{jt} has to be defined. This event is defined as "measurement j originated from target t". Since it is desired that the pdf of each false measurement is uniformly distributed in the entire surveillance region, no gate is used, *i.e.*, $P_G = 1$. This simplifies the expressions of the events, but it increases the computational burden. Logic is used to avoid considering events whose probabilities are negligible, thereby limiting the required computational effort.
The logic includes the derivation of a binary validation matrix Ω. The rows of Ω represent the measurements. All rows have a first element equal to one, representing that the measurement can result from clutter. The remaining columns are used to indicate, for each of the measurements, whether it falls within a validation region, *i.e.*, the second column has unit value for all measurements that fall within validation region one, the third column has unit value for all measurements that fall within validation region two, and so on. To clarify this, the validation matrix corresponding to Figure 3.2 is given under the assumption that there are two targets within the surveillance region

$$\Omega = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}. \tag{B.6}$$

In this Ω, the first column shows that all measurements can be false, the second column shows that measurements z_1, z_2, z_3, and z_4 fall within the first validation region and the third column shows that z_4, z_5, z_7, and z_8 fall within the second validation region. Please note that z_6 falls outside both validation regions and, therefore, row six only contains the standard unit value in the first column.
In reality, each target can only generate one measurement, and each measurement can only originate from one target. Obviously, the number of false measurements is not limited. A matrix that keeps this restriction in mind is constructed and named $\hat{\Omega}$, *i.e.*, it is a validation matrix that refers to feasible events only. This means that all rows an columns of $\hat{\Omega}$ can only contain one unit value (with exception of the first column representing the false measurements).
With some definitions that arise naturally during the construction of $\hat{\Omega}$ and using Bayes' law, probabilities of joint events are derived. Then, together with the above mentioned assumption that the measurement associated with a target has a normal distribution, the final formulations can be derived. To prevent bogging down in details, these expressions are not given. However, as mentioned before, these can be found in Bar-Shalom & Fortmann [1988]. Most important property of the final expressions of the JPDAF is that the continuation of the track is chosen to be a weighted average of all validated measurements, where the probabilities are the weights.

Appendix C

Details Tracking Related Methods

C.1 Linear Kalman filter

The linear Kalman filter (LKF) gives the optimal Bayesian solution to the state estimation problem, if the posterior pdf at every time step is Gaussian. If $p(x_{k-1}|z_{1:k-1})$ is Gaussian, this assumption holds if f_k and h_k in Equations (2.1) and (2.2) are known and linear in x_{k-1} and v_{k-1}, respectively x_k and w_k. Furthermore, v_{k-1} and w_k should be drawn from zero mean Gaussian distributions with known covariances Q_k and R_k. Then, equations (2.1) and (2.2) can be written as:

$$x_k = F_k x_{k-1} + v_{k-1} \tag{C.1}$$
$$z_k = H_k x_k + w_k, \tag{C.2}$$

where F_k and H_k are the transition, respectively, the observation model. In Equation (C.1), the term $G_k u_{k-1}$ can be included to represent a (known) control input signal u_k and the control input model G_k, however, this does not influence the results presented in the sequel of this section and therefore, it is omitted. More details about the steps to be presented can be found in Bar-Shalom & Fortmann [1988], or in the famous paper Kalman [1960]. More information about Gaussian (sometimes called normal) distributions can be found in, *e.g.*, Calvetti & Somersalo [2007].
The prediction $\hat{x}_k$ of the state x_k at time k is calculated using

$$\hat{x}_{k|k-1} = F_k \hat{x}_{k-1|k-1}. \tag{C.3}$$

The error $\tilde{x}_{k|k-1} \triangleq x_k - \hat{x}_{k|k-1}$ now becomes

$$\tilde{x}_{k|k-1} = F_{k-1} \tilde{x}_{k-1|k-1} + v_{k-1}, \tag{C.4}$$

and the one step prediction covariance $P_{k|k-1} \triangleq E(\tilde{x}_{k|k-1} \tilde{x}^T_{k|k-1})$ is

$$P_{k|k-1} = F_{k-1} P_{k-1|k-1} F^T_{k-1} + Q_{k-1}. \tag{C.5}$$

Furthermore, the prediction $\hat{z}_{k|k-1}$ of the measurement z_k is calculated using

$$\hat{z}_{k|k-1} = H_k \hat{x}_{k|k-1}. \tag{C.6}$$

The error $\tilde{z}_{k|k-1} \triangleq z_k - \hat{z}_{k|k-1}$ becomes

$$\tilde{z}_{k|k-1} = H_k \tilde{x}_{k|k-1} + w_k, \tag{C.7}$$

and the measurement prediction covariance $S_k \triangleq E(\tilde{z}_{k|k-1}\tilde{z}^T_{k|k-1})$ is

$$S_k = H_k P_{k|k-1} H_k^T + R_k. \tag{C.8}$$

Using, these expressions, the Kalman gain can be derived resulting in

$$K_k = P_{k|k-1} H_k^T S_k^{-1}. \tag{C.9}$$

After defining the innovation $\nu_k \triangleq \tilde{z}_{k|k-1}$, calculated as

$$\nu_k = z_k - H_k \hat{x}_{k|k-1}, \tag{C.10}$$

the state estimate can be written as

$$x_{k|k} = \hat{x}_{k|k-1} + K_k \nu_k, \tag{C.11}$$

which is called the state update equation. Note that the measurement prediction covariance by definition corresponds to the innovation covariance. The state covariance can be written as

$$P_{k|k} = P_{k|k-1} - K_k H_k P_{k|k-1}. \tag{C.12}$$

All the steps that are taken during one state estimation cycle are summarized in Figure C.1. This representation is based on Bar-Shalom & Fortmann [1988, Figure 2-4].
An important advantage of the LKF is that it only requires the storage of a small amount of information (the mean and the variance of the distribution). This results in a good compromise between the computational complexity and representational flexibility.

C.2 Sequential Monte Carlo methods

This section introduces sequential Monte Carlo (SMC) sampling methods. These methods can be used to do approximations, *e.g.*, high dimensional complex integrals. They can be used either in a Bayesian and a non-Bayesian framework. This section focuses on SMC methods in the context of the target tracking problem. The explanation that is given in this appendix is based on Doucet *et al.* [2001], Doucet *et al.* [2000], and Bishop [2006]. For more details is referred to these references.
Monte Carlo (MC) is defined by "approximating an expectation by the sample mean of a function of simulated random variables", according to Anderson [1999]. If N i.i.d. samples $\{x^{(i)}_{0:k};\, i = 1, \ldots, N\}$ can be simulated according to the posterior pdf $p(x_{0:k}|z_{1:k})$, then a straightforward approximation of this pdf is:

$$P_N(dx_{0:n}|z_{1:n}) = \frac{1}{N} \sum_{i=1}^{N} \delta_{x^{(i)}_{0:n}} dx_{0:n}. \tag{C.13}$$

The expectation for some function of interest f_k is:

$$I(f_n) = \int f_n(x_{0:n}) p(x_{0:n}|z_{1:n}) dx_{0:n}. \tag{C.14}$$

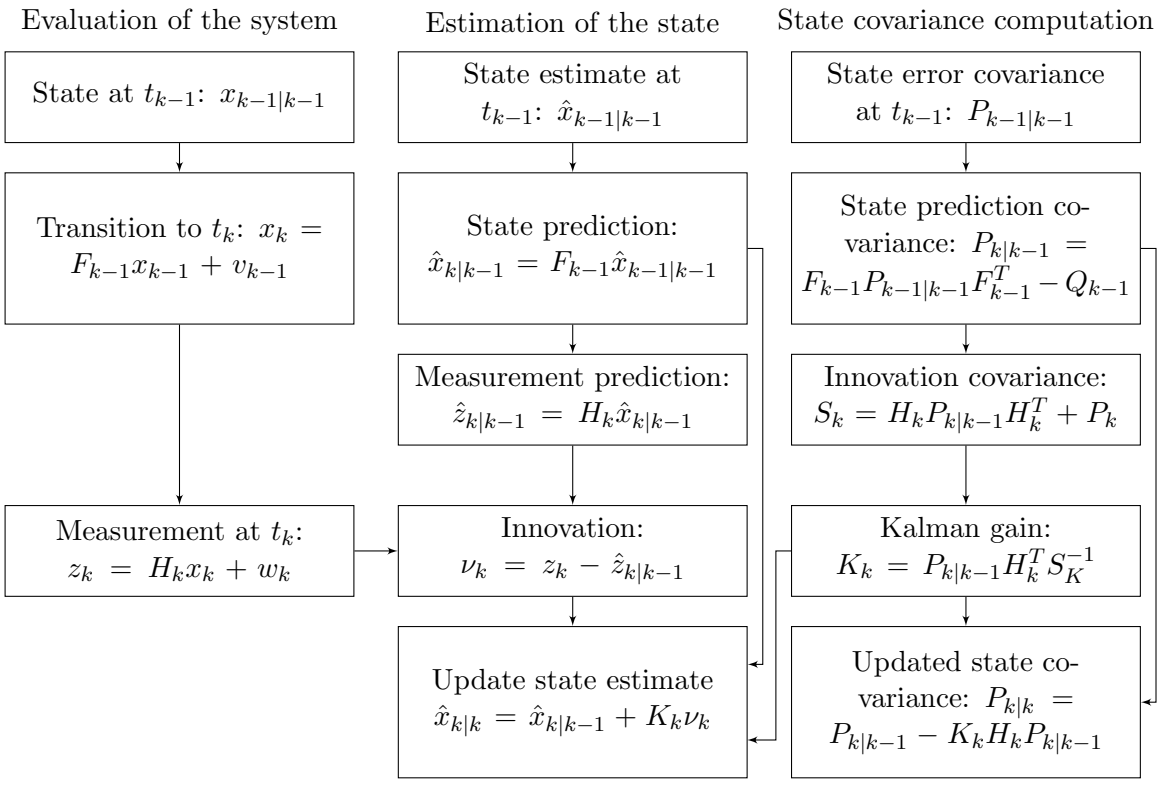

Figure C.1: One cycle of the state estimation of a linear system using a Kalman filter.

An obvious choice for the approximation of $I(f_k)$ would be using Equation (C.13) instead of the real pdf, leading to:

$$I_N(f_n) = \frac{1}{N}\sum_{i=1}^{N} f_k(x_{0:n}^{(i)}). \tag{C.15}$$

Using the strong law of large numbers and the central limit theorem, it can be shown that this approximation is unbiased and has appealing convergence properties. Unfortunately, efficient sampling from the posterior distribution is not possible. Markov chain Monte Carlo (MCMC) methods allow sampling, but because of the iterative algorithms involved, these methods are unsuited to recursive estimation problems such as real time tracking. For that reason, an importance sampling (IS) approach is adopted. With this approach, the goal is to take samples mainly from regions that are important, *i.e.*, that have a high probability. IS is based on drawing samples from a normalized importance sampling distribution $\pi(x_{0:k}|z_{1:k})$, also called importance or proposal function. This importance function should be as similar to the probability distribution as possible. It should at least not be small in regions where the probability is large. Quite often, the posterior distribution is known up to a normalizing constant, since the denominator of Bayes' rule is difficult to calculate. In case of IS, the MC estimate of $I(f_k)$ is

$$\hat{I}_N(f_n) = \sum_{i=1}^{N} f_n(x_{0:n}^{(i)})\tilde{w}_n^{(i)}, \tag{C.16}$$

where the required normalizing important weights $\tilde{w}_n^{(i)}$ are given by

$$\tilde{w}_n^{(i)} = \frac{w(x_{0:n}^{(i)})}{\sum_{j=1}^{N} w(x_{0:n}^{(j)})}, \tag{C.17}$$

and therein, the importance weight $w(x_{x_{0:n}}^{(i)})$ is given by

$$w(x_{x_{0:n}}^{(i)}) = \frac{p(x_{0:n}|z_{1:n})}{\pi(x_{0:n}|z_{1:n})}. \tag{C.18}$$

A more detailed derivation of this result can be found in, *e.g.*, Bishop [2006] or Doucet *et al.* [2001].
This method can be interpreted as a sampling method where the posterior distribution $p(x_{0:k}|z_{1:k})$ is approximated by

$$\hat{P}_N(dx_{0:k}|z_{1:k}) = \frac{1}{N}\sum_{i=1}^{N} \tilde{w}^{(i)}\delta_{x_{0:n}^{(i)}}\, dx_{0:n}. \tag{C.19}$$

Unfortunately, the computational complexity of this method increases in time, since at each time step, all data up to time k is used. This problem is solved by the sequential IS (SIS) algorithm, where the importance function is chosen to be of the form

$$\pi(x_{0:n}|z_{1:n}) = \pi(x_0)\prod_{k=1}^{n} \pi(x_k|x_{0:k-1},\ z_{1:k}). \tag{C.20}$$

Such an importance function allows the evaluation of the importance weights recursively in time. If the importance function is chosen to be equal to the prior distribution, the following expressions can be derived:

$$\pi(x_{0:n}|z_{1:n}) = p(x_0) \prod_{k=1}^{n} p(x_k|x_{k-1}), \tag{C.21}$$

and

$$\tilde{w}_k^{(i)} \propto \tilde{w}_{k-1}^{(i)} p(z_k|\, x_k^{(i)}). \tag{C.22}$$

In practice, after a few time steps, all but one particle will have negligible weight. The variance of such an importance function can only increase over time. This problem, called degeneracy problem, usually is tackled by introducing an additional resampling step. Such a resampling step is used in the bootstrap filter, introduced in Gordon *et al.* [1993]. The key idea of the bootstrap filter is to eliminate all particles having a low importance weight and to multiply particles having a high importance weight. This is done such that the total number of particles remains unchanged.
More formally, the bootstrap filter replaces Equation (C.19) by

$$P_N(dx_{0:k}|z_{1:k}) = \frac{1}{N} \sum_{i=1}^{N} N_t^{(i)} \delta_{x_{0:n}^{(i)}} dx_{0:n}, \tag{C.23}$$

where the integer number $N_t^{(i)}$ represents the number of offspring associated to particle $x_{0:n}^{(i)}$. If this number equals zero, a particle dies and the sum of all $N_t^{(i)}$ should be equal to N. $N_t^{(i)}$ is chosen in such a way that

$$\int f_n(x_{0:k}) P_N(dx_{0:k}|z_{1:k}) \approx \int f_n(x_{0:k}) \hat{P}_N(dx_{0:k}|z_{1:k}). \tag{C.24}$$

The surviving particles are used to make a prediction step resulting in variety between the particles. A clear visual representation of the method is given in Figure C.2, reproduced from Doucet *et al.* [2001]. A step by step explanation is as follows:

1. First, N particles are available at time $t-1$. These particles provide an approximation of $p(x_{k-1}|\, z_{1:k-2})$ and are shown on the top line of the figure. The particles are not weighted and in this example $N = 10$.
2. Using the measurement performed at time $k-1$, the particle weights corresponding to the particles introduced at the previous step are calculated. These weights are used to approximate $p(x_{k-1}|\, z_{1:k-1})$, using Equation (C.16). The weighted particles are represented by the black dots with different size right below the first curve.
3. This is the resampling step. Only the fittest particles, *i.e.*, those with highest weight, are given offspring. In this example, four particles die, which means that four other particles are given birth. The new set, still containing $N = 10$ particles is not weighted and still approximates $p(x_{k-1}|\, z_{1:k-1})$.
4. The fourth step is, called prediction step, is where variety is introduced. This step is represented by the arrows roughly halfway of the figure and the result are the particles right above the second curve. The variety is introduced by, *e.g.*, using the model or the transition probability kernel. The resulting set of particles is an approximation of $p(x_k|\, z_{1:k-1})$. Now the situation is as in the first step.

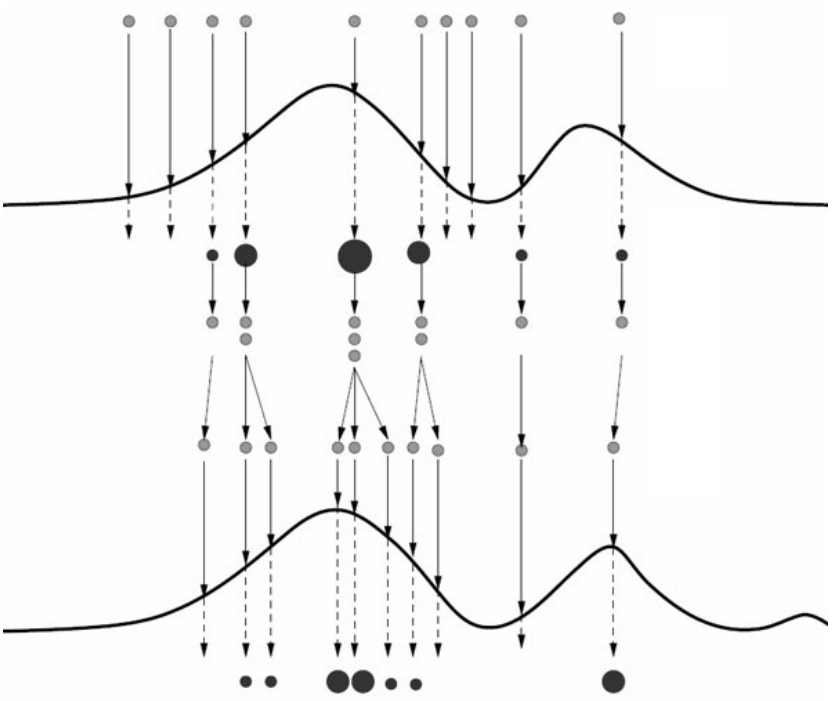

Figure C.2: Visualization of the bootstrap filter, reproduced from Doucet *et al.* [2001].

In the bootstrap filter introduced by Gordon *et al.* [1993], the surviving particles are obtained by sampling N times from $\hat{P}_N(dx_{0:k}|z_{0:k})$, but many alternative ways exists, each with there own computational characteristics, *e.g.*, systematic resampling as in Arulampalam *et al.* [2002]. Advantage of SMC methods is that they are, unlike grid-based methods, very flexible, easy to implement, and applicable in very general settings, according to Doucet *et al.* [2001]. Very important with these type of filters is the selection of the importance function. The most important aspect during the selection of an importance function is that the importance function has to be large in regions where the probability is large in order to allow an efficient implementation. More about the selection of an importance function or SMC methods in general can be found in, *e.g.*, Arulampalam *et al.* [2002], Bishop [2006], or Doucet *et al.* [2000]. Drawback of SMC methods is their relatively high computational complexity.

C.3 Multi-target tracking using the probability hypothesis density

The summary in this section is based on Mahler [2003]. The single-sensor single-target optimal Bayesian solution usually is to complex to be calculated exactly. For that reason, the multi-sensor multi-target solution can only be found using "drastic but intelligent approximation strategies". The strategy in Mahler [2003] is analog to the LKF for single-sensor single-target problems, which means that only the first-order moment statistic, *i.e.*, the expectation, and not the entire multi-target posterior distribution is propagated.

Assuming that the expectation is sufficient to represent the posterior distribution means assuming:

$$f_{k|k}(X_k|Z^k) \cong f(X_k|x_{D|k}), \tag{C.25}$$

where $f_{k|k}(X_k|Z^k)$ is the multi-target posterior density, X_k is the set of multi-target states, Z^k contains all measurements by al sensors at time k, and $D_{k|k}$ is first order statistical moment. In other words, it is assumed that "some first order statistical moment $D_{k|k}$ is an approximate sufficient statistic". $D_{k|k}$ is named the probability hypothesis density (PHD). An important property of the PHD is that it is related to the expected number of objects, as stated in Mahler [2003]. More specifically, the integral of the PHD over any area S is the expected number of targets within S. This approximation is reasonable if:

1. The information loss involving $f_{k|k}(X_k|Z^k)$ by $D_{k|k}$ should be minimized.
2. The prediction $D_{k+1|k}$ based on $D_{k|k}$ should be the first moment of $f_{k+1|k}(X_k|Z^k)$.
3. The corrector up-dating $D_{k+1|k}$ to $D_{k+1|k+1}$ should be lossless.

A discussion regarding these points can be found in the original work Mahler [2003].