Probabilistic Methods in Multiple Target Tracking
–
Review and Bibliography

Research report B36. ISBN 951-22-6938-4.

Simo Särkkä, Toni Tamminen, Aki Vehtari and Jouko Lampinen
Laboratory of Computational Engineering
Helsinki University of Technology
P.O.Box 9203, FIN-02015, HUT, Finland
{Simo.Sarkka,Toni.Tamminen,Aki.Vehtari,Jouko.Lampinen}@hut.fi

February 12, 2004

Abstract

The purpose of this document is to review the most commonly used algorithms in multiple target tracking and data association. All reviewed algorithms belong to class of probabilistic or Bayesian methods, that is, they represent all uncertainties as probability distributions. The purpose of the document is not to be a complete reference of existing multiple target tracking methods, but more like an extended review.

The document also reviews a collection of Monte Carlo methods, which are commonly used in Bayesian inference, and which are useful in multiple target tracking. Because dynamic modeling is also an important issue in practical applications and written material on the topic is quite scattered in the literature, a review of dynamic modeling with stochastic differential equations is also included. The dynamic modeling review is written in practical engineering point of view, without rigorous mathematical treatment of the stochastic differential equations.

Abbreviations and notations

Abbreviations

CLT	Central Limit Theorem
DLM	Dynamic Linear Model
CDLM	Conditional Dynamic Linear Model
EKF	Extended Kalman Filter
FD	Finite Differences
FEM	Finite Element Method
FPK	Fokker-Planck-Kolmogorov
FPKE	Fokker-Planck-Kolmogorov Equation
HMC	Hybrid Monte Carlo
IID	Independently and Identically Distributed
IMM	Interacting Multiple Model
JPDA	Joint Probabilistic Data Association
JPDAC	JPDA with Coupled Targets
JPDAM	JPDA with Merged Measurements
KF	Kalman Filter
LTI	Linear Time Invariant
LQG	Linear Quadratic Gaussian
MC	Monte Carlo
MCDA	Monte Carlo Data Association
MCJPDA	Monte Carlo Joint Probabilistic Data Association
MCMC	Markov Chain Monte Carlo
MHT	Multiple Hypothesis Tracking
MMSE	Minimum Mean Squared Error
ODE	Ordinary Differential Equation
PDA	Probabilistic Data Association
PDF	Probability Density Function
RBPF	Rao-Blackwellized Particle Filter
SDE	Stochastic Differential Equation
SIR	Sequential Importance Resampling
SIS	Sequential Importance Sampling

SMC	Sequential Monte Carlo
UKF	Unscented Kalman Filter
UPF	Unscented Particle Filter
UT	Unscented Transformation

Symbols

$A(\cdot\|\cdot)$	Acceptance probability of Metropolis-Hastings
A	Sensor's surveillance area
$\mathbf{A}$	Temporary matrix in matrix fraction expansion
$\mathbf{A}_k$	Feedback matrix of discrete model on time jump $t_{k-1} \to t_k$
$\mathbf{A}_{j,k}$	Feedback matrix of discrete model of target j on time jump $t_{k-1} \to t_k$
$\mathbf{A}(t)$	Temporary time dependent matrix in matrix fraction expansion
$\mathbf{A}(\mathbf{x}, \mathbf{q})$	Derivative of feedback function with respect to state
a	Angular velocity
$\mathbf{a}(\cdot)$	Feedback function in discrete state space model
$\mathbf{B}$	Temporary matrix in matrix fraction expansion
$\mathbf{B}(t)$	Temporary time dependent matrix in matrix fraction expansion
$\mathbf{D}$	Set of measurement data
$\mathbf{D}^{(1)}$	First coefficient of Fokker-Planck-Kolmogorov Equation
$\mathbf{D}^{(2)}$	Second coefficient of Fokker-Planck-Kolmogorov Equation
$\delta(\tau)$	Dirac delta function
Δt_k	Time difference $t_{k+1} - t_k$
$\mathbf{e}_i$	Vector, where element i is 1, others 0.
$E[h(\mathbf{x})]$	Expected value of function $h(\mathbf{x})$ when $\mathbf{x}$ is distributed as $p(\mathbf{x})$
$E[h(\mathbf{x})\|\mathbf{y}]$	Expected value of function $h(\mathbf{x})$ when $\mathbf{x}$ is distributed as $p(\mathbf{x}\|\mathbf{y})$
$E(\cdot)$	Energy Function
$\mathbf{f}(\cdot)$	Feedback function of dynamical model (ODE)
$\mathrm{F}_u(\cdot)$	Generic update function
$\mathrm{F}_p(\cdot)$	Generic prediction function
$\mathrm{F}_{lh}(\cdot)$	Generic likelihood function
$\mathbf{F}$	Feedback matrix of LTI model
$\mathbf{F}(t)$	Feedback matrix of linear model
$\mathbf{g}(\mathbf{x})$	Generic function $\mathbf{g} : \mathbb{R}^n \to \mathbb{R}^d$
$g(\mathbf{x})$	Generic function $g : \mathbb{R}^n \to \mathbb{R}$
$G_j(\mathbf{x})$	Gaussian distribution tied to target j
$\mathbf{H}_k$	Linear measurement model matrix on time t_k
$\mathbf{H}_{j,k}$	Linear measurement model matrix of target j on time t_k
$\mathbf{h}(\cdot)$	Measurement function in (discrete) state space model
$\mathbf{H}(\mathbf{x}, \mathbf{r})$	Derivative of measurement function with respect to state
$\mathbf{I}$	Identity Matrix
$\mathbf{L}$	Process noise matrix in LTI model

$\mathbf{L}(t)$	Process noise matrix in linear model
$L(\mathbf{x})$	Likelihood function
λ_k	Latent variable of time step t_k in MHT
λ_k	Latent variable of time step k in Rao-Blackwellized Filter
Λ_k	Set of latent variables from steps $t_1, \ldots, t_k$
$\mathbf{K}_k$	Kalman gain
$\mathrm{KF}_\mathrm{p}(\cdot)$	Kalman filter prediction function
$\mathrm{KF}_\mathrm{u}(\cdot)$	Kalman filter update function
$\mathrm{KF}_\mathrm{lh}(\cdot)$	Kalman filter measurement likelihood function
κ_k	Boolean indicator of positive/negative measurement at time step k
$\mathbf{m}_k$	State mean on time t_k
$\mathbf{m}_k^-$	State mean on time t_k just before measurement $\mathbf{y}_k$
$\mathbf{m}_k^{(i)}$	State mean in particle i on time t_k after measurement $\mathbf{y}_k$
$\mathbf{m}_k^{-(i)}$	State mean in particle i on time t_k just before measurement $\mathbf{y}_k$
$\mathbf{m}_{j,k}^{(i)}$	Target j state mean in particle i on time t_k after measurement $\mathbf{y}_k$
$\mathbf{m}_{j,k}^{-(i)}$	Target j state mean in particle i on time t_k just before measurement $\mathbf{y}_k$
$\mathbf{m}(t)$	State mean as continuous function of time
M	Number of modes in IMM
N	Number of particles
$N(\cdot)$	Gaussian distribution
$O(\cdot)$	$g(N) \in O(f(N))$ means that there exists $c_0, c_1, N_0 > 0$ such that $N > N_0$ implies $g(N) < c_0 + c_1 f(N)$
$o(\cdot)$	$g(h) \in o(f(h))$ means that there exists $c_0, c_1, N_0 > 0$ such that $N > N_0$ implies $g(N) > c_0 + c_1 f(N)$
P_{err}	Probability that sensor fails to detect a target
$\mathbf{P}$	State covariance
$\mathbf{P}_k$	State covariance of time t_k
$\mathbf{P}_k^-$	State covariance of time t_k just before measurement $\mathbf{y}_k$
$\mathbf{P}_k^{(i)}$	State covariance in particle i of time t_k after measurement $\mathbf{y}_k$
$\mathbf{P}_k^{-(i)}$	State covariance in particle i of time t_k just before measurement $\mathbf{y}_k$
$\mathbf{P}_{j,k}^{(i)}$	Target j state covariance in particle i of time t_k after measurement $\mathbf{y}_k$
$\mathbf{P}_{j,k}^{-(i)}$	Target j state covariance in particle i of time t_k before measurement $\mathbf{y}_k$
$\mathbf{P}(t)$	State covariance as continuous function of time
$\tilde{p}$	Fourier Transformation of p
$\mathbf{p}$	Vector (or set) of probabilities
p_c	Prior probability of clutter
p_j	Prior probability of target j association, where $j = 0$ means clutter
$p(\mathbf{x})$	Prior or marginal distribution of $\mathbf{x}$
$p(\mathbf{y}\|\mathbf{x})$	Probability (likelihood) of data $\mathbf{y}$ given state $\mathbf{x}$
$p(\mathbf{x}\|\mathbf{D})$	Posterior distribution of state $\mathbf{x}$ given data $\mathbf{D}$
$\mathbf{p}(t)$	Parameters of dynamical model
$\pi(\cdot)$	Importance distribution in Importance Sampling

Π	Set of importance distributions for particles in particle set
q	Scalar parameter defining the strength of spectral density
$\mathbf{q}_k$	Process noise on time jump $t_{k-1} \to t_k$
$\mathbf{q}_{j,k}$	Process noise of target j on time jump $t_{k-1} \to t_k$
$\mathbf{Q}_c$	Spectral density
$\mathbf{Q}_c(t)$	Time dependent Spectral density
$\mathbf{Q}_k$	Process covariance on time jump $t_{k-1} \to t_k$
$\mathbf{Q}_{j,k}$	Process covariance of target j on time jump $t_{k-1} \to t_k$
$Q(\cdot\|\cdot)$	Proposal distribution in Metropolis-Hastings
r	Radius
$\dot{r}$	Radial velocity
$\mathbf{r}_k$	Measurement noise of measurement k on time t_k
$\mathbf{r}_{j,k}$	Target j measurement noise of measurement k on time t_k
$\mathbf{R}_k$	Covariance of measurement noise on time step t_k
$\mathbf{R}_{j,k}$	Covariance of target j measurement noise on time step t_k
$\mathbb{R}$	Space of real numbers
$\mathbb{R}^n$	Space of real vectors of dimension n
ρ	Temporary scalar variable
ρ_m	Weight of mode m in IMM
$\mathbf{S}$	Non-centered covariance, i.e. correlation
$\mathbf{S}_k$	Innovation covariance in Kalman filter
Σ	Covariance of Gaussian Distribution
θ	Angle
$\dot{\theta}$	Angular velocity
T	Number of targets
t	Independent variable representing time
t_k	Time step t_k, time of measurement k
$\mathrm{tr}(\mathbf{A})$	Trace of matrix $\mathbf{A}$, sum of diagonal elements
τ	Independent variable representing time
u	Uniformly distributed random scalar
$U(\cdot)$	Uniform Distribution
V	Volume of clutter measurement space
$\mathbf{v}_k$	Kalman filter innovation on step k
$\mathbf{V}_k$	Noise derivative of measurement model
$\mathbf{V}(\mathbf{x}, \mathbf{r})$	Derivative of measurement function with respect to noise
$\mathbf{w}(t)$	Continuous white noise process
$w_k^{(i)}$	Importance weight of particle i on time step k in SIR
W_i	Sigma weight in Unscented Transformation
$\mathbf{W}(\mathbf{x}, \mathbf{q})$	Derivative of feedback function with respect to noise
$\mathbf{W}_{k-1}$	Noise matrix of discrete dynamical model on time jump $t_{k-1} \to t_k$
$\mathcal{W}_k$	Set of particle weights on time step k after measurement k
$\mathcal{W}_k^-$	Set of particle weights on time step k just before measurement k

ω	Independent variable in Fourier Transformation
$\mathbf{x}(t)$	State at time t
$\mathbf{x}_k$	State at time t_k
$\mathbf{x}_k^{(i)}$	Sample i in Monte Carlo representation of states at time t_k
$\mathbf{x}_{j,k}$	State of target j at time t_k
$\mathbf{x}_{j,k}^{(i)}$	State of target j in particle i at time t_k
$\mathbf{x}^*$	Candidate state in MCMC methods
$\mathbf{x}_{-i}$	Vector $\mathbf{x}$ without component i
$\dot{\mathbf{x}}$	Derivative of $\mathbf{x}$ with respect to time t
$\ddot{\mathbf{x}}$	Second derivative of $\mathbf{x}$ with respect to time t
$\mathbf{X}_k$	Stacked vector of states of all targets at time step k
$\mathcal{X}_k$	Set of state particles on time step k
$\mathbf{X}_i$	Sigma point in Unscented Transformation
$\mathcal{X}$	Space of states $\mathbf{x}$
$\mathbf{y}_k$	Measurement at time step t_k
$\mathbf{y}_{j,k}$	Measurement at time step from target j
$\mathbf{y}_{1:k}$	Collected measurement data until time t_k
$\mathbf{y}_{j,1:k}$	Collected measurement data until time t_k from target j
$\overline{y}$	Mean of $\mathbf{y}$
$\mathbf{y}_s$	Measurement from specific sensor s
$\mathcal{Y}$	Space of measurements $\mathbf{y}$
Z	Normalization factor
$\sim$	$\mathbf{x} \sim p(\mathbf{x})$ means that random variable $\mathbf{x}$ is distributed as $p(\mathbf{x})$
$\approx$	$\mathbf{x} \approx \mathbf{y}$ means that $\mathbf{x}$ is approximately $\mathbf{y}$
$\propto$	$p(\mathbf{x}) \propto f(\mathbf{x})$ means that $p(\mathbf{x}) = f(\mathbf{x})/Z$ for some constant Z.
$\partial\mathbf{f}/\partial\mathbf{x}$	Jacobian matrix of vector valued function $\mathbf{f}(\mathbf{x})$
$\nabla f(\mathbf{x})$	Gradient of scalar valued function $f(\mathbf{x})$
$\nabla^T f(\mathbf{x})$	Transpose of gradient of scalar valued function $f(\mathbf{x})$
$\nabla\nabla^T f(\mathbf{x})$	Hessian matrix of scalar valued function $f(\mathbf{x})$
$\forall j$	For all values of j

Contents

List of Algorithms

Chapter 1

Introduction

1.1 Overview of the Problem

The basic tracking scenario consists of *sensors*, which produce noisy measurements, for example, azimuth angle measurements as illustrated in Figure 1.1. The purpose of tracking algorithm is to determine the *target trajectory* using the sensor measurements. There is additional *prior information* on the dynamics of targets, which restricts the forms of target trajectories into those that are possible when the laws of physics are taken into account.

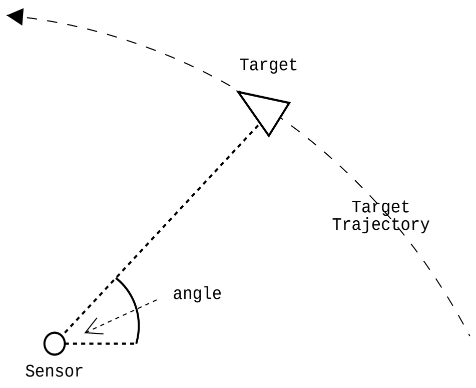


Figure 1.1: *Sensor (depicted as circle) generates angle measurements of the target (depicted as triangle), and the purpose is to determine the target trajectory (dashed line).*

More general tracking scenario consists of *multiple sensors*, which may have different precisions, and they can produce different kinds of measurements, also other than azimuth angle measurements. Typically, sensors are not synchronized and they produce measurements during irregular intervals. Figure 1.2 illustrates the case of multiple sensors. In estimation point of view, increasing number of sensors will ease the estimation procedure, since we get more information on the same target trajectory.

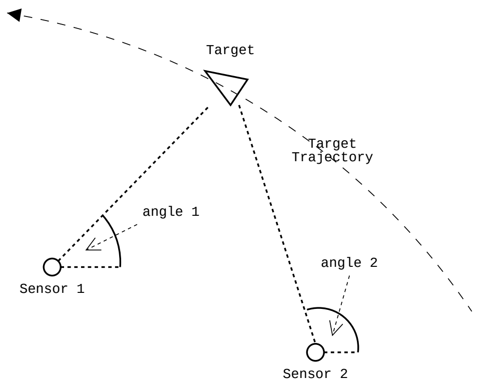


Figure 1.2: *Multiple sensors give us more information on the same trajectory and thus ease the estimation procedure.*

In case of multiple targets there is an additional difficulty, because without additional information we do not know which of the measurements correspond to which targets. Figure 1.3 illustrates this problem – if the observed information are the angle measurements 1 – 4, how do we know which targets they belong to? This is called as the problem of *data association*. The same problem applies to *false alarm or clutter measurements*, since we do not know if a given measurement was false alarm or measurement from a target.

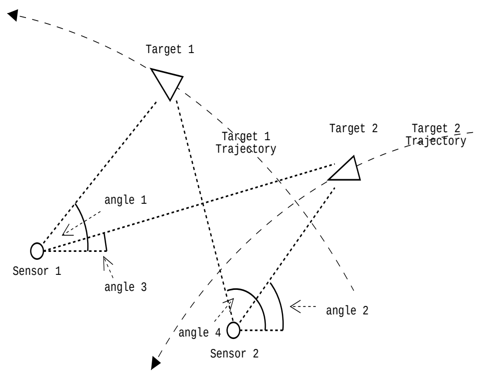


Figure 1.3: *In case of multiple targets, it is impossible to know without any additional information, which target produced which measurement.*

Attribute measurements can be used for inferring properties of the targets. For example, the attribute measurements might contain information that target is most likely a fighter. Estimation algorithm may then use the dynamic model for the correct platform type instead of averaging over all possible platform types according to their prior (or step-back posterior) probabilities. Attributes also implicitly help the procedure of data association, because if the platform type that was inferred from measured attributes doesn't match with the prior estimate of platform type, the association likelihood will be lower and other way around.

Negative information arises in situation, when we can somehow infer that a sensor didn't give measurement from a target, which is assumed to exist. An artificial *negative information measurement* is then generated, which is a measurement indicating that target is likely to be outside the surveillance area of the sensor.

1.2 Bayesian Inference

The mathematical approach in this document is based on philosophy that we formulate and solve the estimation problems entirely on probabilistic basis[1]. Estimation methods are developed for estimating the whole probability distributions as accurately as possible, and individual parameter estimates are derived from these distributions. Thus, we don't derive estimators for parameters, but for probability distributions.

The fundamental theorem of non-linear estimation and filtering theory is that posterior distribution of states contains all possible information on state that can be derived from measurements and model (Jazwinski, 1970; Bar-Shalom et al., 2001). In data association case this theorem states that joint posterior distribution of the states and associations contains all possible information that can be derived from measurements, given the model.

Bayesian inference (see, e.g., Bernardo and Smith, 1994; Gelman et al., 1995) utilizes the theory of statistics for building mathematical models of the nature. It provides intuitive approach for deriving models, which are statistically optimal, such that it is not even in theory possible to make better models in statistical sense, conditional to modeling assumptions. Of course, the modeling assumptions are never perfectly accurate and so the model will always be sub-optimal. Additional sub-optimality is due to approximations required for converting theoretical computational models into algorithms suitable for sequential processing in digital computers.

The procedure of Bayesian inference is in principle the following:

1. Carefully formulate the *states* and *measurements* involved in the phenomenon including their values, ranges and connections to physical world. In this document, the unknown states (such as position and velocity of a vehicle)

[1]Pure probabilistic methods are often called as Bayesian methods

are denoted as

$$\mathbf{x} \in \mathcal{X}, \tag{1.1}$$

where $\mathcal{X} \subset \mathbb{R}^n$ is the space of states. The measurements are denoted as

$$\mathbf{y} \in \mathcal{Y}, \tag{1.2}$$

where $\mathcal{Y} \subset \mathbb{R}^m$ is the space of measurements.

2. Formulate *forward model* or *noise model*, which connects the hidden states to measurements, taking the uncertainty of measurement model into account. For example, if correct position was known, what would we measure in average and what would be the distribution of error. This model is called the likelihood:

   $$p(\mathbf{y}|\mathbf{x}) = \{\text{probability of measurement } \mathbf{y} \text{ given state } \mathbf{x}\}. \tag{1.3}$$

   Note that likelihood means the probability density here, not the logarithm of it, unlike within many texts on classical statistics for example in book Milton and Arnold (1995).

3. Combine multiple measurements into *joint likelihood*:

   $$p(\mathbf{y}_1, \ldots, \mathbf{y}_T|\mathbf{x}) = \prod_t p(\mathbf{y}_t|\mathbf{x}). \tag{1.4}$$

   The posterior probability density is now given as

   $$p(\mathbf{x}|\mathbf{y}_1, \ldots, \mathbf{y}_T) \propto p(\mathbf{x}) \prod_t p(\mathbf{y}_t|\mathbf{x}), \tag{1.5}$$

   where $\propto$ means *proportional* to such that left hand side is constant times the right hand side. Note that in case of probability densities this constant can be always calculated since probability densities must integrate to unity with respect to the random variable.

   In fact the expanded formula is

   $$p(\mathbf{x}|\mathbf{y}_1, \ldots, \mathbf{y}_T) = \frac{p(\mathbf{x}) \prod_t p(\mathbf{y}_t|\mathbf{x})}{\int p(\mathbf{x}) \prod_t p(\mathbf{y}_t|\mathbf{x}) \, d\mathbf{x}}. \tag{1.6}$$

   where the denominator is independent of $\mathbf{x}$.

4. Formulate *prior model* for states $\mathbf{x}$, which can for example constrain the velocities and position into sensible certain area. This prior model can be also *non-informative*[2], if no additional prior information is available.

[2] for example, uniform over all possible values

Another types of prior models are *dynamical models* such as stochastic differential equations in form

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{w}(t)), \tag{1.7}$$

which can be thought as prior models that define the time dependent stochastic behavior of states. Here $\dot{\mathbf{x}}$ stands for derivative of $\mathbf{x}$ with respect to time t and $\mathbf{w}(t)$ is the *process noise* modeling the uncertainties in dynamics.

5. Develop *estimation algorithms* for calculating the expectations (which are known to be *Minimum Mean Squared Error*, MMSE estimators) in form

$$\mathrm{E}[\mathbf{g}(\mathbf{x})] = \int \mathbf{g}(\mathbf{x}) \; p(\mathbf{x} | \mathbf{y}_1, \ldots, \mathbf{y}_T) \, \mathrm{d}\mathbf{x}. \tag{1.8}$$

   where $\mathbf{g}(\mathbf{x})$ is an arbitrary function. The closed form integration of Equation (1.8) is impossible for almost all practical distributions and functions. However, if all functions involved are linear and noise models are Gaussian, the closed form integration is possible (see e.g. Section 5.2). These linear Gaussian solutions include the pseudo-inverse solution for matrices and Kalman filter for dynamical models. This is one of the reasons for their popularity.

   For nonlinear functions and non-Gaussian distributions, more complex methods are required. These methods include number of numerical integration methods such as Monte Carlo integration (see Chapter 2).

Techniques, where we first formulate accurate forward model, apply the prior information and then invert the combined model are sometimes called *stochastic inversion* algorithms. One way of formulating this kind of inversion problem is using Bayesian analysis techniques, as we do in this document.

1.3 Filtering and State Estimation

Filtering in this context is equivalent to *optimal state estimation in dynamic environment*. The aim is to estimate hidden signal consisting of sequence of states, using indirect noisy measurements. Estimation is *optimal* in sense that it minimizes the expected estimation error. In case of multiple targets, minimizing the estimation error also requires that we are able to solve the problem of data association, with possible help of attribute measurements or without. Estimation is done on-line, which means that updated state estimate is available as soon as new measurement has been obtained. In Bayesian sense filtering is equivalent to *recursive estimation of posterior distribution of states*.

Tracking is form of filtering where the hidden signal consists of states of a moving or stationary targets (position, velocity etc.). The purpose of tracking is to estimate the position of these targets as accurately as possible using only indirect and noisy measurements such as azimuths, distances and elevations. In addition to these kinematic measurements there are attribute measurements, which help process of data association by providing information on types of targets in area. In order to reach the goal of optimality, the estimation algorithm should be complete in sense that it is able to track all existing targets, and clutter insensitive such that there are no extra targets in addition to existing ones.

Measurement model defines how much we can infer from indirect measurements. Given a measurement, we could in theory calculate estimate of the state by inverting this measurement model. But unfortunately, using the measurements alone, we could only calculate an approximate position for the exact time instant when new measurement arrived, but not for any other time instants. The information in previous measurements would be always lost, because there is no *time bridge* between measurements.

Measurement model is typically in form

$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k, \mathbf{r}_k), \tag{1.9}$$

where $\mathbf{h}(\cdot)$ is an arbitrary but known function from state $\mathbf{x}$ to measurement $\mathbf{y}$ and k denotes the measurement number. This function actually defines what we would measure if we knew the state and there was no noise (e.g., $\mathbf{r}_k = 0$). The model also contains the noise term $\mathbf{r}_k$ which is used for modeling the uncertainty in real measurements.

Dynamical model tries build a "time bridge" between the measurements of different times. It is typically based on physical laws (e.g., differential equations), which loosely define the target's most likely movements. For example, if the target is an airplane, it should obey physical laws. This means that, for example, it cannot change direction very fast, and it must have certain velocity to keep above ground.

Typically dynamical model is presented in form of *stochastic differential equation*

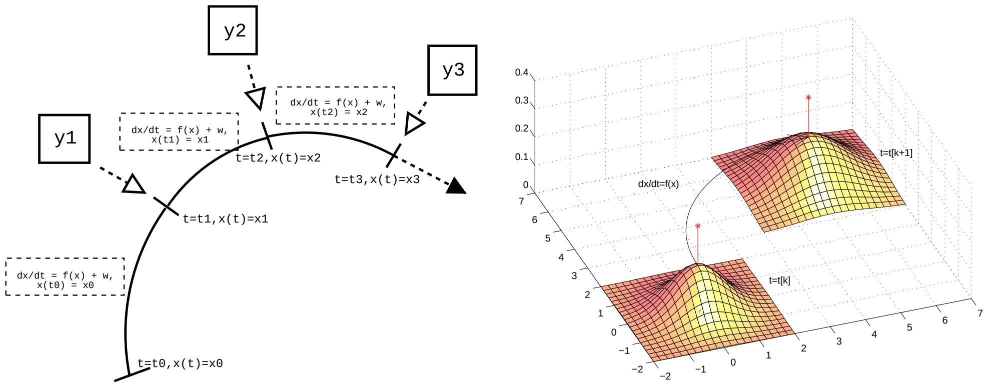
$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{w}(t)), \tag{1.10}$$

where $\mathbf{w}(t)$ is an unknown noise term. In practice, we integrate Equation (1.10) from measurement to measurement. Therefore dynamical model in Equation (1.10) can be also written as *discrete jump Markov model*, jumping from measurement to measurement:

$$\mathbf{x}_{k+1} = \mathbf{f}_k(\mathbf{x}_k, \mathbf{q}_k). \tag{1.11}$$

Filter is an algorithm, which does the job of optimal state estimation using the measurement and dynamical models defined in Equations (1.9) and (1.10) respectively. The basic functionality of a filter is the following:

1. **Initialization:** Start from initial state $\mathbf{x}(t_0) = \mathbf{x}_0$ with suitable degree of uncertainty.
2. **Prediction:** When a new measurement arrives, *use the dynamical model to predict* where the measurement should be, when only the old measurements are taken into account. Prediction step is visualized in Figure 1.4.
3. **Update:** *Use the new measurement to update* the state estimate a bit so that we have optimal combination of old and new information, weighted according to their quality (probability). Update step is visualized in Figure 1.5.

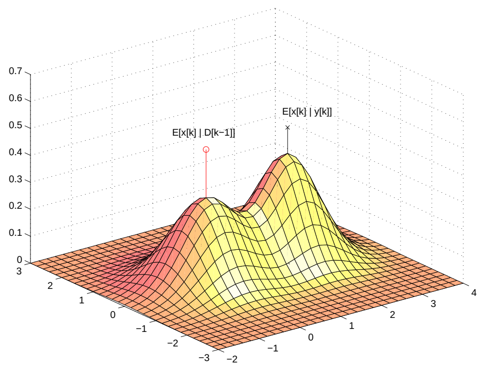


(a) (b)

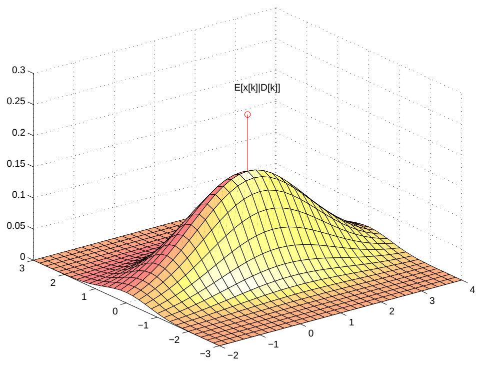
Figure 1.4: *(a) Filtering can be thought as integration of dynamics from measurement to measurement and updating the state estimate using the measurements. (b) Dynamical model is used for integrating both the state estimate and its uncertainty from measurement to measurement.*

1.4 Optimal Filtering

The success of *optimal linear filtering* is mostly due to the journal paper of Kalman (1960), which describes a recursive solution to the discrete linear filtering problem. Although, the original derivation of *Kalman filter* was based on least squares approach, the same equations can be derived from pure probabilistic Bayesian analysis. The Bayesian analysis of Kalman filtering is well covered in the classic book of Jazwinski (1970) and more recently in the book of Bar-Shalom et al. (2001).



(a)



(b)

Figure 1.5: *(a) When measurement arrives, we have two differing estimates – the realized noisy measurement and predicted measurement which both have their uncertainties. (b) Update step combines the predicted and realized measurements in optimal fashion depending on their uncertainty.*

Kalman filtering, mostly because of its least squares interpretation, has been widely used in stochastic optimal control. A practical reason to this is that the inventor of Kalman filter, Rudolph E. Kalman, has also made several contributions to the theory of *linear quadratic Gaussian* (LQG) regulators (see, e.g., Maybeck, 1982b), which are fundamental tools of stochastic optimal control.

As discussed in the book of West and Harrison (1997), in the sixties, Kalman filter type recursive estimators were also used in Bayesian community and it is not clear if theory of Kalman filtering or theory of *dynamic linear models* (DLM) was the first. Although, these theories were originally derived from slightly different starting points, they are equivalent. This document approaches the Bayesian filtering problem in Kalman filtering point of view, because of its useful connection to the theory and history of stochastic optimal control.

In the early stages of its history, Kalman filter was soon discovered to belong to the class of Bayesian estimators (see, e.g., Ho and Lee, 1964), and the resulting generalized theory is called as *non-linear filtering theory* (Jazwinski, 1970). This theory is fundamentally Bayesian, because there really is no other way of formulating the non-linear theory in a complete and mathematically rigorous manner. An interesting historical detail is that while Kalman and Bucy were formulating the linear theory in the United States, Stratonovich was doing the pioneering work on the probabilistic (Bayesian) approach in Russia (Jazwinski, 1970).

1.5 Continuous-Discrete Filtering

Optimal discrete filter, such as Kalman filter, solves the discrete filtering problem, which means that the underlying physical phenomenon is modeled as a discrete time process. However, because the nature is continuous, physically more realistic approach is *continuous-discrete filtering*, where the state dynamics are modeled as continuous time stochastic processes (i.e., stochastic differential equations) and measurements are assumed to be obtained during discrete time steps. This continuous-discrete filtering approach is due to Jazwinski (1970).

In theory, an optimal continuous-discrete filter does the following (Jazwinski, 1970):

- *Prediction step* solves the probability density at time step t_k from *Fokker-Planck-Kolmogorov (FPK) partial differential equation* using the old posterior distribution at time step t_{k-1} as the boundary condition.
- *Update step* fuses the information in new measurement at time step t_k with the prior information contained in the predicted posterior distribution above using the *Bayes' rule*.

When we implement the above steps for specific model, we rarely need to consider the actual theoretical basis in algorithms. For example, the continuous-discrete Kalman filter does exactly the above, but due to linearity these steps reduce to:

- *Prediction step* solves the mean and covariance from *ordinary differential equations* describing the dynamics of these moments.
- *Update step* calculates the new state mean and covariance using the Kalman filter update equations.

As shown in Section 3.6 the above procedure can be actually implemented as pure discrete process, because the solution to mean and covariance propagation equations is actually linear function of initial conditions.

Chapter 2

Monte Carlo Methods

2.1 Principles and Motivation of Monte Carlo

Within statistical methods in engineering and science it is often necessary to evaluate expectations in form

$$E[\mathbf{g}(\mathbf{x})] = \int \mathbf{g}(\mathbf{x})\ p(\mathbf{x})\ \mathrm{d}\mathbf{x}. \tag{2.1}$$

where $\mathbf{g} : \mathbb{R}^n \to \mathbb{R}^m$ in an arbitrary function and $p(\mathbf{x})$ is the probability distribution (density) of $\mathbf{x}$.

This is especially the case within methods of estimation theory since the posterior expectation is the MMSE estimator of such quantity. Unfortunately, closed evaluation of such expectations turns out to be impossible in all except in few cases such as linear Gaussian case. Therefore, numerical methods have been applied in majority of the cases.

Monte Carlo methods provide a numerical method of calculating the integrals in form of Equation (2.1). Monte Carlo refers to general class of methods, where closed form computation of statistical quantities is replaced by drawing samples from the distribution and estimating the quantities by sample averages.

In (perfect) Monte Carlo approximation, we draw independent random samples from $\mathbf{x}^{(i)} \sim p(\mathbf{x})$ and estimate expectation as

$$E[\mathbf{g}(\mathbf{x})] \approx \frac{1}{N} \sum_i \mathbf{g}(\mathbf{x}^{(i)}). \tag{2.2}$$

Monte Carlo methods approximate the target density by number of samples that are distributes according to the target density. Figure 2.1 represents a two dimensional Gaussian distribution and its Monte Carlo representation.

The convergence of Monte Carlo approximation is guaranteed by Central Limit Theorem (CLT) (see, e.g., Liu, 2001) and the error term is $O(N^{-1/2})$, regardless of dimensionality of $\mathbf{x}$. This invariance of dimensionality is unique to

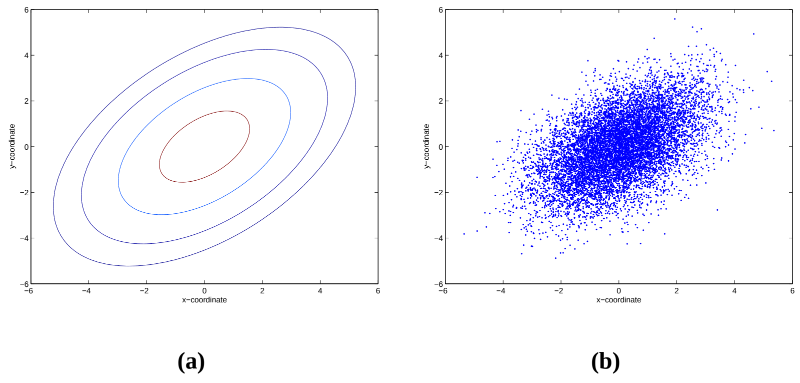


Figure 2.1: (a) Two dimensional Gaussian density. (b) Monte Carlo representation of the same Gaussian density.

Monte Carlo methods and makes them superior to practically all other numerical methods when dimensionality of $\mathbf{x}$ is considerable.

Importance Sampling (see, Section 2.2) is a simple algorithm for generating *weighted* samples from the target distribution. The difference to the perfect Monte Carlo sampling is that each of the particles contain a weight, which corrects the difference between actual target density and the approximation obtained from importance distribution. Importance sampling is very important, since it is the basis of *Sequential Importance Sampling* (see, Section 5.5), which in turn is the basis of all Particle Filtering algorithms.

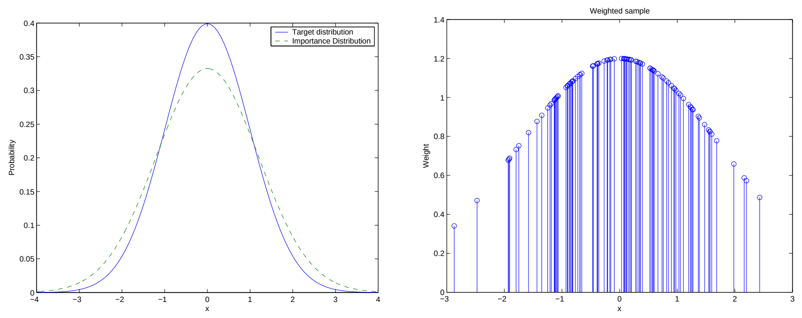
2.2 Importance Sampling

It is not always possible to obtain samples directly from $p(\mathbf{x})$ due to its complex form. In *Importance Sampling* (see, e.g., Liu, 2001) we use approximate distribution called importance distribution $\pi(\mathbf{x})$ from which we can draw samples. Having samples $\mathbf{x}^{(i)} \sim \pi(\mathbf{x})$ we could approximate the expectation integral (2.1) as

$$E[\mathbf{g}(\mathbf{x})] \approx \frac{1}{N} \sum_i \frac{\mathbf{g}(\mathbf{x}^{(i)})\ p(\mathbf{x}^{(i)})}{\pi(\mathbf{x}^{(i)})}. \tag{2.3}$$

Figure 2.2 illustrates the idea of Importance sampling. We sample from the importance distribution, which is an approximation to target distribution. Since the distribution of samples is not the correct one, we have to associate a weight to each of the samples in order to make the distribution correct.

Disadvantage is that we should be able to evaluate $p(\mathbf{x}^{(i)})$ in order to use it di-



(a) **(b)**

Figure 2.2: (a) Importance distribution approximates the target distribution (b) Weights are associated to each of the samples to correct the approximation.

rectly, but we often don't know the normalization constant of $p(\mathbf{x}^{(i)})$. *Importance Sampling* method uses an approximation, where we define weights as

$$w_i = \frac{p(\mathbf{x}^{(i)})}{\pi(\mathbf{x}^{(i)})}. \tag{2.4}$$

It approximates the expectation as

$$E[\mathbf{g}(\mathbf{x})] \approx \frac{\sum_i \mathbf{g}(\mathbf{x}^{(i)})\, w_i}{\sum_i w_i}, \tag{2.5}$$

which has the fortunate property that we don't have to know the normalization constant of $p(\mathbf{x})$.

2.3 Metropolis-Hastings

Metropolis-Hasting algorithm (Metropolis et al., 1953; Hastings, 1970) forms the basis of all *Markov Chain Monte Carlo* (MCMC) methods (see, e.g., Gilks et al., 1996). It can be used for generating samples from distribution $p(\mathbf{x})$, which is known up to normalization constant. At first the algorithm draws initial sample point $\mathbf{x}^{(0)}$ from some approximate distribution. Then the steps described in Algorithm 1 are repeated until sufficient number of samples has $\mathbf{x}^{(i)}$ been obtained.

We can prove the correctness of algorithm by noting that the Markov chain of accepted points, which has transition kernel

$$K(\mathbf{x}, \mathbf{x}^*) = A(\mathbf{x}^*|\mathbf{x})\, Q(\mathbf{x}^*|\mathbf{x}), \tag{2.7}$$

1. Let $\mathbf{x}$ be the current sample point. Draw new candidate point $\mathbf{x}^*$ from proposal distribution $Q(\mathbf{x}^*|\mathbf{x})$.
2. Accept the candidate point with probability

$$A(\mathbf{x}^*|\mathbf{x}) = \min\left\{\frac{p(\mathbf{x}^*)Q(\mathbf{x}|\mathbf{x}^*)}{p(\mathbf{x})Q(\mathbf{x}^*|\mathbf{x})}, 1\right\}. \tag{2.6}$$

In practice we should first draw number from uniform distribution $u \sim U(0,1)$. If $u < A(\mathbf{x}^*|\mathbf{x})$ then store $\mathbf{x}^*$ and let $\mathbf{x} \leftarrow \mathbf{x}^*$. Otherwise reject the candidate point.

Algorithm 1: Metropolis-Hastings

has $p(\mathbf{x})$ as its stationary distribution. This is due to fact that detailed balance condition is met, which implies that Markov chain is reversible in time:

$$\begin{aligned} p(\mathbf{x})A(\mathbf{x}^*|\mathbf{x})Q(\mathbf{x}^*|\mathbf{x}) &= \min\left\{p(\mathbf{x})Q(\mathbf{x}^*|\mathbf{x}),\, p(\mathbf{x}^*)Q(\mathbf{x}|\mathbf{x}^*)\right\} \\ &= \min\left\{p(\mathbf{x}^*)Q(\mathbf{x}|\mathbf{x}^*),\, p(\mathbf{x})Q(\mathbf{x}^*|\mathbf{x})\right\} \\ &= p(\mathbf{x}^*)A(\mathbf{x}|\mathbf{x}^*)Q(\mathbf{x}|\mathbf{x}^*). \end{aligned} \tag{2.8}$$

Various forms of proposal distribution can be used:

1. **Symmetric Proposal Distribution:** Markov chain kernel $K(\mathbf{x}, \mathbf{x}^*)$ is called symmetric if $K(\mathbf{x}, \mathbf{x}^*) = K(\mathbf{x}^*, \mathbf{x})$. In *Metropolis algorithm* the proposal distribution depends only on distance between current and candidate point $|\mathbf{x} - \mathbf{x}^*|$, which means that $Q(\mathbf{x}^*|\mathbf{x}) = Q(\mathbf{x}|\mathbf{x}^*)$. Then Metropolis-Hastings acceptance probability reduces to

$$A(\mathbf{x}^*|\mathbf{x}) = \min\left\{p(\mathbf{x}^*)/p(\mathbf{x}), 1\right\}. \tag{2.9}$$

2. **Random Walk:** Markov chain, which produces samples if form $\mathbf{x}_{n+1} = \mathbf{x}_n + w_n$, where w_n is independent of chain, is called Random Walk. In proposal distribution this means that its value depends only on difference of current and candidate point $Q(\mathbf{x}^*|\mathbf{x}) = p_w(\mathbf{x}^* - \mathbf{x})$, where p_w is the probability density of w_n.

3. **Independence Chains:** In this case candidate probabilities are independent of current point $Q(\mathbf{x}^*|\mathbf{x}) = g(\mathbf{x}^*)$. Function g should be chosen to be as close as possible to the target distribution $p(\mathbf{x})$.

4. **Other Methods:** The most common special case of the algorithm is the *Gibbs Sampling* (section 2.4). Also *Hybrid Monte Carlo (HMC)* (Duane

et al., 1987) can be interpreted as variation of Metropolis-Hastings algorithm.

2.4 Gibbs Sampling

Gibbs sampling (see, e.g., Gilks et al., 1996) can be applied to a distribution $p(\mathbf{x})$ for which we draw exact samples from conditional densities of components given the other components. During one iteration step each of the m components are updated one at a time by drawing new values for them from the conditional densities. Gibbs sampling step is presented as Algorithm 2.

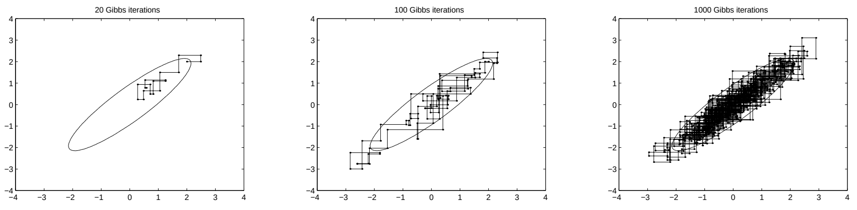


Figure 2.3: Gibbs sampling draws samples from target distribution by sampling from conditional densities of individual components.

On each step, draw new vector $\mathbf{x}_{n+1}$ using the equations below and store the result:

$$\begin{aligned}
\mathbf{x}_1^{(n+1)} &\sim p(\mathbf{x}_1|\mathbf{x}_2^{(n)},\mathbf{x}_3^{(n)},\ldots,\mathbf{x}_m^{(n)})\\
\mathbf{x}_2^{(n+1)} &\sim p(\mathbf{x}_2|\mathbf{x}_1^{(n+1)},\mathbf{x}_3^{(n)},\ldots,\mathbf{x}_m^{(n)})\\
\mathbf{x}_3^{(n+1)} &\sim p(\mathbf{x}_3|\mathbf{x}_1^{(n+1)},\mathbf{x}_2^{(n+1)},\mathbf{x}_4^{(n)},\ldots,\mathbf{x}_m^{(n)})\\
&\ldots\\
\mathbf{x}_m^{(n+1)} &\sim p(\mathbf{x}_m|\mathbf{x}_1^{(n+1)},\mathbf{x}_2^{(n+1)},\ldots,\mathbf{x}_{m-1}^{(n+1)})
\end{aligned}$$

Algorithm 2: Gibbs Sampling

Gibbs sampling can be interpreted as special case of Metropolis-Hastings algorithm. Let's assume that updates are done one at a time and Metropolis-Hastings step is done after each update. If the probability of candidate component generated by i:th update is $p(\mathbf{x}^*|\mathbf{x}_{-i}^{(n)})$ then the acceptance probability is

$$\begin{aligned}
A(\mathbf{x}^*|\mathbf{x}^{(n)}) &= \min\left\{\frac{p(\mathbf{x}^*)\,Q(\mathbf{x}^{(n)}|\mathbf{x}^*)}{p(\mathbf{x}^{(n)})\,Q(\mathbf{x}^*|\mathbf{x}^{(n)})}, 1\right\} \\
&= \min\left\{\frac{p(\mathbf{x}^*)\,p(\mathbf{x}_i^{(n)})|\mathbf{x}_{-i}^{(n)})}{p(\mathbf{x}^{(n)})\,p(\mathbf{x}_i^*|\mathbf{x}_{-i}^{(n)})}, 1\right\} \\
&= \min\left\{\frac{p(\mathbf{x}_{-i}^n)}{p(\mathbf{x}_{-i}^n)}, 1\right\} \\
&= \min\{1, 1\} = 1,
\end{aligned}$$

which means that every proposal is accepted.

Chapter 3

Solutions to Linear Stochastic Differential Equations

3.1 State Space Form of Ordinary Differential Equations

The purpose of this chapter is to familiarize reader to using *stochastic differential equations* as dynamic models. The most attention is targeted to *linear models*, which are the most common types of dynamic models used in tracking problems.

At first, illustration purposes, we will derive a simple differential equation model and present it in *state space form*. Presenting ordinary differential equations in state space form means that higher order differential equation is represented as system of first order differential equations in form (see, e.g., Kreyszig, 1993)

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, t). \tag{3.1}$$

For example, second order differential equation

$$\ddot{x} = 0 \tag{3.2}$$

can be converted into state space form by defining variables as follows:

$$x_1 = x \tag{3.3}$$

$$x_2 = \dot{x}. \tag{3.4}$$

Now equation (3.2) can be rewritten as

$$\dot{x}_1 = x_2 \tag{3.5}$$

$$\dot{x}_2 = 0. \tag{3.6}$$

This can be written in vectorial form (state space form)

$$\dot{\mathbf{x}} = \mathbf{F}\mathbf{x}, \tag{3.7}$$

where

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \qquad \mathbf{F} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}. \tag{3.8}$$

The purpose of this chapter is *not* to be a reference of existing dynamic models. The idea is to demonstrate how dynamic models can be converted into the generic form (state space form), which can be used in estimation algorithms.

In stochastic case, the most general differential equation concerned here is the Langevin equation (Stratonovich, 1968; Jazwinski, 1970; Risken, 1989)

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, t) + \mathbf{L}(\mathbf{x}, t)\mathbf{w}(t), \tag{3.9}$$

where $\mathbf{w}(t)$ is the noise component. Our derivation will actually lead to stochastic differential equations in Stratonovich sense, because we are using ordinary calculus for manipulating the integrals. Equivalent Ito equations could be obtained by using suitable transformation rules (Stratonovich, 1968; Jazwinski, 1970)

3.2 Linear Motion Models

Target dynamics in tracking problems are often modeled as *Linear Time Invariant Ordinary Differential Equations* in form

$$\dot{\mathbf{x}} = \mathbf{F}\mathbf{x} + \mathbf{L}\mathbf{w}(t), \tag{3.10}$$

where $\mathbf{w}(t) \sim N(\mathbf{0}, \mathbf{Q}_c)$ is the (diagonal) process noise, which models the uncertainty in dynamics. This is not actually Ordinary Differential Equation but *Stochastic Ordinary Differential Equation* because of the noise component.

Let's first consider a simple Linear Time Invariant (LTI) model, which is the constant velocity or equivalently zero acceleration model:

$$\ddot{x} = 0 \tag{3.11}$$
$$\ddot{y} = 0, \tag{3.12}$$

where $\ddot{x}$ denotes the second time derivative of x-coordinate, that is, x-directional acceleration. The model given by Equations (3.11) and (3.12) states that all feasible target trajectories should have exactly zero acceleration everywhere. Which is, of course, unnatural assumption. For this reason, we introduce process noises $w_x(t)$ and $w_y(t)$ as follows:

$$\ddot{x} = w_x(t) \qquad w_x(t) \sim N(0, q_x) \tag{3.13}$$
$$\ddot{y} = w_y(t) \qquad w_y(t) \sim N(0, q_y). \tag{3.14}$$

This model allows trajectories that may vary from exactly zero acceleration motion, but still are most likely close to zero acceleration in short periods of time.

The model above is in form of Equation (3.10), which can be seen by rewriting this model in form

$$\dot{x} = \dot{x} \tag{3.15}$$
$$\dot{y} = \dot{y} \tag{3.16}$$
$$\ddot{x} = w_x(t) \tag{3.17}$$
$$\ddot{y} = w_y(t) \tag{3.18}$$

If we define the state as $\mathbf{x} = (x\ y\ \dot{x}\ \dot{y})^T$ and noise as $\mathbf{w} = (w_x\ w_y)$, this model is in form of Equation (3.10) with

$$\mathbf{F} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \qquad \mathbf{L} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad \mathbf{Q}_c = \begin{pmatrix} q_x & 0 \\ 0 & q_y \end{pmatrix}. \tag{3.19}$$

Another example of LTI model is the constant angular acceleration model

$$\ddot{x} = -a\dot{y} + w_x(t) \qquad w_x(t) \sim N(0, q_x) \tag{3.20}$$
$$\ddot{y} = a\dot{x} + w_y(t) \qquad w_y(t) \sim N(0, q_y), \tag{3.21}$$

where a is the known constant angular acceleration. This model is also in form of Equation (3.10) with

$$\mathbf{F} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -a \\ 0 & 0 & a & 0 \end{pmatrix} \qquad \mathbf{L} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad \mathbf{Q}_c = \begin{pmatrix} q_x & 0 \\ 0 & q_y \end{pmatrix}. \tag{3.22}$$

Linear Differential Equations may also depend on time, in which case the model can be written in form

$$\dot{\mathbf{x}} = \mathbf{F}(t)\mathbf{x} + \mathbf{L}(t)\mathbf{w}(t). \tag{3.23}$$

An example of linear but *time dependent* model is the known angular velocity model, where the angular velocity $a(t)$ changes in deterministic way. The model in this case is

$$\ddot{x} = -a(t)\dot{y} + w_x(t) \qquad w_x(t) \sim N(0, q_x) \tag{3.24}$$
$$\ddot{y} = a(t)\dot{x} + w_y(t) \qquad w_y(t) \sim N(0, q_y), \tag{3.25}$$

which can be written in form of Equation (3.23) by defining

$$\mathbf{F}(t) = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -a(t) \\ 0 & 0 & a(t) & 0 \end{pmatrix} \qquad \mathbf{L} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad \mathbf{Q}_c = \begin{pmatrix} q_x & 0 \\ 0 & q_y \end{pmatrix}. \tag{3.26}$$

Linear Differential Equations have the property that they are linear operators, which implies that the stochastic linear differential Equation (3.23) transforms Gaussian initial conditions into Gaussian distributions over any time period. Non-linear differential equations will transform Gaussian initial conditions into non-Gaussian distributions (see, e.g., Jazwinski, 1970).

3.3 Nearly Constant Angular Velocity Model

Consider non-linear *nearly constant angular velocity model* or *coordinated turn model* in form

$$\ddot{x} = -a\dot{y} + w_1(t) \tag{3.27}$$

$$\ddot{y} = a\dot{x} + w_2(t) \tag{3.28}$$

$$\dot{a} = w_3(t), \tag{3.29}$$

where

$$w_1(t) \sim N(0, q_x) \tag{3.30}$$

$$w_2(t) \sim N(0, q_y) \tag{3.31}$$

$$w_3(t) \sim N(0, q_a). \tag{3.32}$$

The model is in form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{L}\mathbf{w}(t), \tag{3.33}$$

where the state is defined as $\mathbf{x} = (x\ y\ \dot{x}\ \dot{y}\ a)^T$ and

$$\mathbf{f}(\mathbf{x}) = \begin{pmatrix} \dot{x} \\ \dot{y} \\ -a\dot{y} \\ a\dot{x} \\ 0 \end{pmatrix} \qquad \mathbf{L} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad \mathbf{Q}_c = \begin{pmatrix} q_x & 0 & 0 \\ 0 & q_y & 0 \\ 0 & 0 & q_a \end{pmatrix}. \tag{3.34}$$

This model is non-linear, because function $\mathbf{f}$ is non-linear.

3.4 Solutions of Stochastic Linear Differential Equations

Stochastic linear differential equations in form

$$\dot{\mathbf{x}}(t) = \mathbf{F}(t)\mathbf{x}(t) + \mathbf{L}(t)\mathbf{w}(t), \tag{3.35}$$

where initial conditions are

$$\mathbf{x}(0) = N(\mathbf{m}(0), \mathbf{P}(0)), \tag{3.36}$$

$\mathbf{F}(t)$ and $\mathbf{L}(t)$ are matrix valued functions, and $\mathbf{w}(t)$ is a Gaussian white noise process with spectral density $\mathbf{Q}_c$, can be solved exactly using ordinary differential equations

$$\dot{\mathbf{m}}(t) = \mathbf{F}(t)\mathbf{m}(t) \tag{3.37}$$

$$\dot{\mathbf{P}}(t) = \mathbf{F}(t)\mathbf{P}(t) + \mathbf{P}(t)\mathbf{F}(t)^T + \mathbf{L}(t)\mathbf{Q}_c(t)\mathbf{L}(t)^T. \tag{3.38}$$

The latter of these is called as the *Lyapunov differential equation* and is special case of *Riccati differential equation*. There equations are the classical optimal prediction equations of Kalman-Bucy filter (see; e.g., Kalman and Bucy, 1961; Jazwinski, 1970; Stengel, 1994; Bar-Shalom et al., 2001).

3.5 Solutions of Stochastic Linear Time Invariant Differential Equations

Time independent versions of stochastic linear differential equations, which are called as *stochastic linear time invariant differential equations*, are very useful in dynamic modeling. They have the general form

$$\dot{\mathbf{x}}(t) = \mathbf{F}\mathbf{x}(t) + \mathbf{L}\mathbf{w}(t), \tag{3.39}$$

where initial conditions are

$$\mathbf{x}(0) = N(\mathbf{m}(0), \mathbf{P}(0)), \tag{3.40}$$

$\mathbf{F}$ and $\mathbf{L}$ are constant matrices, and $\mathbf{w}(t)$ is a Gaussian white noise process with moments

$$\mathrm{E}[\mathbf{w}(t)] = 0 \tag{3.41}$$

$$\mathrm{E}[\mathbf{w}(t)\, \mathbf{w}(t+\tau)^T] = \mathbf{Q}_c\delta(\tau). \tag{3.42}$$

Mean Equation (3.37) has now closed form solution, which can be written in terms of matrix exponential

$$\mathbf{m}(t) = \exp\{\mathbf{F}t\}\mathbf{m}(0). \tag{3.43}$$

The corresponding Lyapunov equation can be solved by using matrix fractions (see, e.g., Stengel, 1994). If we define matrices $\mathbf{A}$ and $\mathbf{B}$ such that $\mathbf{P} = \mathbf{A}\mathbf{B}^{-1}$, it is easy to show that $\mathbf{P}$ solves the Lyapunov Equation (3.38) if matrices $\mathbf{A}$ and $\mathbf{B}$ solve the differential equation

$$\begin{pmatrix} \dot{\mathbf{A}} \\ \dot{\mathbf{B}} \end{pmatrix} = \begin{pmatrix} \mathbf{F} & \mathbf{L}\mathbf{Q}_c\mathbf{L}^T \\ \mathbf{0} & -\mathbf{F}^T \end{pmatrix} \begin{pmatrix} \mathbf{A} \\ \mathbf{B} \end{pmatrix}, \tag{3.44}$$

and $\mathbf{P}(0) = \mathbf{A}(0)\mathbf{B}(0)^{-1}$. We can select, for example,

$$\mathbf{A}(0) = \mathbf{P}(0) \tag{3.45}$$
$$\mathbf{B}(0) = \mathbf{I}. \tag{3.46}$$

Because differential equation (3.44) is linear and time invariant, it can be solved using the matrix exponential function:

$$\begin{pmatrix} \mathbf{A}(t) \\ \mathbf{B}(t) \end{pmatrix} = \exp\left\{ \begin{pmatrix} \mathbf{F} & \mathbf{L}\mathbf{Q}_c\mathbf{L}^T \\ \mathbf{0} & -\mathbf{F}^T \end{pmatrix} t \right\} \begin{pmatrix} \mathbf{A}(0) \\ \mathbf{B}(0) \end{pmatrix}. \tag{3.47}$$

Equivalent, but more explicit solution to Lyapunov Equation (3.38) in time invariant case is given as

$$\begin{aligned} \mathbf{P}(t) &= \exp(\mathbf{F}t)\mathbf{P}(0)\exp(\mathbf{F}t)^T \\ &\quad + \int_0^t \exp(\mathbf{F}(t-\tau))\mathbf{L}\mathbf{Q}_c(\tau)\mathbf{L}^T \exp(\mathbf{F}(t-\tau))^T \mathrm{d}\tau. \end{aligned} \tag{3.48}$$

However, solution of Equation (3.47) is easier to use by numerical methods.

3.6 From Continuous Models to Discrete Markov Models

Assume that dynamical model is Linear Time Invariant (LTI) model in form

$$\dot{\mathbf{x}}(t) = \mathbf{F}\mathbf{x}(t) + \mathbf{L}\mathbf{w}(t), \tag{3.49}$$

with initial conditions

$$\mathbf{x}(t_0) \sim N(\mathbf{m}(t_0), \mathbf{P}(t_0)), \tag{3.50}$$

and we wanted to create corresponding discrete Markov model jumping from time instance t_0 to t_1, then t_1 to t_2 and so on. State $\mathbf{x}(t)$ could be, for example:

$$\mathbf{x} = \begin{pmatrix} x\text{-coordinate} \\ y\text{-coordinate} \\ x\text{-velocity} \\ y\text{-velocity} \end{pmatrix}. \tag{3.51}$$

Using the results from Section 3.5, with constant $\mathbf{Q}_c$ and known initial conditions $\mathbf{m}(t_k)$ and $\mathbf{P}(t_k)$, the mean and covariance of solution can be expressed as

$$\mathbf{m}(t_k + \Delta t_k) = \exp(\mathbf{F}\Delta t_k)\mathbf{m}(t_k) \tag{3.52}$$

$$\begin{aligned} \mathbf{P}(t_k + \Delta t_k) &= \exp(\mathbf{F}\Delta t_k)\mathbf{P}(t_k)\exp(\mathbf{F}\Delta t_k)^T \\ &\quad + \int_0^{\Delta t_k} \exp(\mathbf{F}(\Delta t_k - \tau))\mathbf{L}\mathbf{Q}_c\mathbf{L}^T \exp(\mathbf{F}(\Delta t_k - \tau))^T \mathrm{d}\tau. \end{aligned} \tag{3.53}$$

Due to time-invariance property of LTI models, the Equations (3.52) and (3.53) can be applied recursively. If we discretize as follows:

$$\mathbf{m}_k = \mathbf{m}(t_k) \tag{3.54}$$

$$\mathbf{P}_k = \mathbf{P}(t_k) \tag{3.55}$$

$$\mathbf{A}_k = \exp(\mathbf{F}\Delta t_k) \tag{3.56}$$

$$\mathbf{Q}_k = \int_0^{\Delta t_k} \exp(\mathbf{F}(\Delta t_k - \tau))\mathbf{L}\mathbf{Q}_c\mathbf{L}^T \exp(\mathbf{F}(\Delta t_k - \tau))^T \,\mathrm{d}\tau, \tag{3.57}$$

where we have let $\Delta t_k = t_{k+1} - t_k$, we can integrate the solution to Equation (3.49) exactly using the relations

$$\mathbf{m}_{k+1} = \mathbf{A}_k \mathbf{m}_k \tag{3.58}$$

$$\mathbf{P}_{k+1} = \mathbf{A}_k \mathbf{P}_k \mathbf{A}_k^T + \mathbf{Q}_k. \tag{3.59}$$

This formulation is particularly useful in case of Kalman filter (Kalman, 1960), because the canonical form of Kalman Filter has this kind of discrete dynamic model. The conclusion is that it doesn't matter that Kalman Filter was originally designed for discrete models, it still is exact for linear continuous dynamical models with discrete measurements.

Chapter 4

Solutions to Non-Linear Stochastic Differential Equations

4.1 Theories of Ito and Stratonovich

There are actually two mathematical theories related to stochastic differential equations. The original theory of stochastic calculus was developed by Ito in the forties. Another theory was developed later by Stratonovich (1968). Practical, engineering oriented description of the differences in these formalisms is given in the book of Jazwinski (1970).

In our concern, the main difference in Ito's and Stratonovich's definitions is that Stratonovich integral can treated as conventional integral when it comes to integration rules. Ito's definition requires that one should always use the specific rules of stochastic calculus, because the normal rules don't apply. Ito's definition is more general than that of Stratonovich's, and every Stratonovich integral has equivalent Ito integral.

For the simple reason that calculations are easier with Stratonovich's definition, it is (at least to our knowledge) more widely used in statistical physics (see, e.g., Risken, 1989). For this same reason we shall also use Stratonovich's definition of stochastic integral, and so we can treat the noise components in our equations as functions in classical sense. Note that Jazwinski (1970) always uses Ito's definition of stochastic integral, because there are special cases in continuous-time non-linear filtering, where Stratonovich's theory is not enough. These cases don't arise in our continuous-discrete filtering case.

4.2 Fokker-Planck-Kolmogorov Equations

Langevin equation is a stochastic differential equation in form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, t) + \mathbf{L}(\mathbf{x}, t)\mathbf{w}(t), \tag{4.1}$$

where $\mathbf{w}(t)$ is a Gaussian white noise process with moments

$$\mathrm{E}[\mathbf{w}(t)] = 0 \tag{4.2}$$
$$\mathrm{E}[\mathbf{w}(t)\ \mathbf{w}(t+\tau)^T] = \delta(\tau). \tag{4.3}$$

Note that a noise process with generic positive definite matrix $\mathbf{Q}_c(t)$ can be used by redefining $\mathbf{L}(\mathbf{x}, t)$ as

$$\hat{\mathbf{L}}(\mathbf{x}, t) = \mathbf{L}(\mathbf{x}, t)\mathbf{S}(t), \tag{4.4}$$

where $\mathbf{Q}_c(t) = \mathbf{S}(t)\mathbf{S}(t)^T$.

In this section we shall use *Einstein's summation convention*, which means that we should sum over every index, which appears twice in a product term. Greek letters are used for denoting indices, which should never be summed over. Using this convention, Langevin Equation (4.1) can be equivalently written in form

$$\dot{x}_i = f_i(\mathbf{x}, t) + L_{ij}(\mathbf{x}, t) w_j(t), \tag{4.5}$$

where $w(t)_i$ is a Gaussian random process with the following properties:

$$\begin{aligned} E[w_i(t)] &= 0 \\ E[w_i(t) w_j(t')] &= \delta_{ij}\delta(t - t'). \end{aligned} \tag{4.6}$$

If we start from an initial distribution

$$\mathbf{x}(0) \sim p(\mathbf{x}, 0), \tag{4.7}$$

the distribution on later times $p(\mathbf{x}, t)$ is given by the *Fokker-Planck-Kolmogorov Equation* (FPKE), which is partial differential equation in form (see, e.g. Jazwinski, 1970)

$$\frac{\partial p}{\partial t} = -\frac{\partial}{\partial x_i}\left(D_i^{(1)}(\mathbf{x}, t) p\right) + \frac{1}{2}\frac{\partial^2}{\partial x_i \partial x_j}\left(D_{ij}^{(2)}(\mathbf{x}, t) p\right). \tag{4.8}$$

The drift coefficients are given as

$$D_i^{(1)}(\mathbf{x}, t) = f_i(\mathbf{x}, t) + \frac{1}{2} L_{kj}(\mathbf{x}, t)\frac{\partial L_{ij}(\mathbf{x}, t)}{\partial x_k} \tag{4.9}$$
$$D_{ij}^{(2)}(\mathbf{x}, t) = L_{ik}(\mathbf{x}, t) L_{jk}(\mathbf{x}, t). \tag{4.10}$$

Langevin Equation (4.1) is exactly the type, which arises, for example, in coordinate transformations of linear dynamic models. For nearly constant velocity model in polar coordinates (Equation (7.38)) we have

$$D_1^{(1)}(r,\theta,\dot{r},\dot{\theta}) = \dot{r} \tag{4.11}$$
$$D_2^{(1)}(r,\theta,\dot{r},\dot{\theta}) = \dot{\theta} \tag{4.12}$$
$$D_3^{(1)}(r,\theta,\dot{r},\dot{\theta}) = \dot{\theta}^2 r \tag{4.13}$$
$$D_4^{(1)}(r,\theta,\dot{r},\dot{\theta}) = -\frac{2\dot{r}\dot{\theta}}{r} \tag{4.14}$$
$$D_{33}^{(2)}(r,\theta,\dot{r},\dot{\theta}) = q \tag{4.15}$$
$$D_{44}^{(2)}(r,\theta,\dot{r},\dot{\theta}) = \frac{q}{r}, \tag{4.16}$$

and other terms are zero. The Fokker-Planck-Kolmogorov equation is given as

$$\begin{aligned}\frac{\partial p}{\partial t} &= -\frac{\partial}{\partial r}(\dot{r}\,p) - \frac{\partial}{\partial \theta}\left(\dot{\theta}\,p\right) - \frac{\partial}{\partial \dot{r}}\left(\dot{\theta}^2 r p\right) + \\ &+ \frac{\partial}{\partial \dot{\theta}}\left(\frac{2\dot{r}\dot{\theta}}{r}p\right) + \frac{1}{2}\frac{\partial^2}{\partial \dot{r}^2}(qp) + \frac{1}{2}\frac{\partial^2}{\partial \dot{\theta}^2}\left(\frac{q}{r}p\right),\end{aligned} \tag{4.17}$$

which we should solve for $p(r,\theta,\dot{r},\dot{\theta},t)$.

4.3 Approximate Solutions to Non-Linear Equations

Consider generic non-linear Langevin equation in form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},t) + \mathbf{L}(\mathbf{x},t)\mathbf{w}(t), \tag{4.18}$$

where $\mathbf{w}(t)$ is a Gaussian random process with the following properties:

$$\begin{aligned} E[\mathbf{w}(t)] &= \mathbf{0} \\ E[\mathbf{w}(t)\mathbf{w}(t+\tau)^T] &= \mathbf{Q}_c\delta(\tau). \end{aligned} \tag{4.19}$$

There are several alternative ways for approximately solving the Langevin equation or the associated Fokker-Planck-Kolmogorov (FPK) equation:

- **Linearization** is the most common way of approximating the nonlinear dynamics. The idea is to calculate approximate mean solution from the nonlinear noise-free model and then use the linearized model in covariance propagation

$$\begin{aligned} \dot{\mathbf{m}} &= \mathbf{f}(\mathbf{m}, t) && (4.20) \\ \mathbf{F}(t) &= \frac{\partial \mathbf{f}}{\partial \mathbf{x}^T}(\mathbf{m}(t), t) && (4.21) \\ \mathbf{L}(t) &= \mathbf{L}(\mathbf{m}(t), t) && (4.22) \\ \dot{\mathbf{P}} &= \mathbf{F}(t)\mathbf{P} + \mathbf{P}\mathbf{F}^T(t) + \mathbf{L}(t)\mathbf{Q}_c(t)\mathbf{L}^T(t). && (4.23) \end{aligned}$$

This is actually Extended Kalman Filter (EKF) prediction step in it continuous form.

- **Second Order Taylor Series** approximation of i:th component of function $\mathbf{f}(\mathbf{x}, t)$ with respect to mean $\mathbf{m}$ gives

$$\begin{aligned} f_i(\mathbf{x}, t) \approx f_i(\mathbf{m}, t) + \nabla^T f_i(\mathbf{m}, t)(\mathbf{x} - \mathbf{m}) + \\ + \frac{1}{2}(\mathbf{x} - \mathbf{m})^T \nabla\nabla^T f_i(\mathbf{m}, t)(\mathbf{x} - \mathbf{m}), \end{aligned} \quad (4.24)$$

where $\nabla^T f_i(\mathbf{m}, t)$ (which is a row vector) denotes the transpose of gradient and $\nabla\nabla^T f_i(\mathbf{m}, t)$ is the Hessian matrix. From above we get that

$$\mathrm{E}[f_i(\mathbf{x}, t)] \approx f_i(\mathbf{m}, t) + \frac{1}{2}\mathrm{tr}\left\{\nabla\nabla^T f_i(\mathbf{m}, t)\mathbf{P}\right\}. \quad (4.25)$$

Second order approximation to mean propagation equation can be now written in form

$$\dot{\mathbf{m}} = \mathbf{f}(\mathbf{m}, t) + \frac{1}{2}\sum_i \mathrm{tr}\left\{\nabla\nabla^T f_i(\mathbf{m}, t)\mathbf{P}\right\}\mathbf{e}_i, \quad (4.26)$$

where vectors $\mathbf{e}_i$ are unit coordinate vectors such that their i:th element is 1 and all other are 0. This second order Taylor series approximation is widely used in second order extended Kalman filter.

- **Unscented Transformation** (UT) (Julier and Uhlmann, 1995) can be used for approximating the first two moments (mean and covariance) of state distribution. The idea is to integrate set of deterministically chosen sigma points through the dynamics, and estimate the moments from those points.

- **Interacting Multiple Models** (IMM) (see, e.g., Bar-Shalom et al., 2001) is a special case of Multiple Model Kalman Filter algorithms, and it is well suited to approximating non-linear dynamic models. The state distribution is represented as a mixture Gaussian distribution and there is a special Markov model, which approximates the probability flow between the Gaussian components.

- **Monte Carlo Methods** (Risken, 1989) are common methods in approximating the solutions of Partial Differential Equations such as FPKE. The idea is to discretize the Wiener process (or Brownian motion process), which models the noise in Langevin equation. Then we can simulate sample trajectories from the Langevin equation by using random number generator and Euler integration.

- **Numeric Solving of Partial Differential Equation** with, for example, the method of Galerkin (Gunther et al., 1997), Finite Differences (FD) (Challa and Bar-Shalom, 2000) or Generalized Edgeworth Series (Challa et al., 2000).

Chapter 5

Filtering and State Estimation

5.1 Optimal (Bayesian) Filtering

Optimal filtering (see, e.g. Jazwinski, 1970; Ho and Lee, 1964; Maybeck, 1979, 1982a; Bar-Shalom et al., 2001), also called as Bayesian filtering considers state estimation models in form

$$\mathbf{x}_k \sim p(\mathbf{x}_k \mid \mathbf{x}_{k-1}) \tag{5.1}$$

$$\mathbf{y}_k \sim p(\mathbf{y}_k \mid \mathbf{x}_k), \tag{5.2}$$

where $\mathbf{x}_k$ is the unknown hidden state, which is only observable indirectly through measurements $\mathbf{y}_k$. Measurements are corrupted by measurement noise, which has the known distribution given by $p(\mathbf{y}_k \mid \mathbf{x}_k)$. Markov model for state transitions $p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$ models the time evolution of state between the measurements.

The purpose of state estimation is to infer the state $\mathbf{x}_k$ as accurately as possible using the measurements $\mathbf{y}_k$. According to the philosophy of optimal filtering, the fundamental purpose of optimal filter is to form an approximate representation of the posterior distribution of states. More accurate this representation is, more closer the algorithm is to optimal performance.

Using the basic rules of statistics, we can infer the posterior distribution of state step $\mathbf{x}_k$ conditional to all measurements up to that time

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:k}) = p(\mathbf{x}_k \mid \mathbf{y}_1, \ldots, \mathbf{y}_k). \tag{5.3}$$

Recursive equations for calculating the posterior distribution sequentially in recursive manner are called the Bayesian Filtering equations. The recursion starts from initial distribution

$$\mathbf{x}_0 \sim p(\mathbf{x}_0), \tag{5.4}$$

and the successive posteriors can be calculated from the equations

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:k-1}) = \int p(\mathbf{x}_k \mid \mathbf{x}_{k-1})\, p(\mathbf{x}_{k-1} \mid \mathbf{y}_{1:k-1})\, \mathrm{d}\mathbf{x}_{k-1} \tag{5.5}$$

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:k}) = \frac{1}{Z_k} p(\mathbf{y}_k \mid \mathbf{x}_k)\, p(\mathbf{x}_k \mid \mathbf{y}_{1:k-1}), \tag{5.6}$$

where the normalization constant is given as

$$Z_k = \int p(\mathbf{y}_k \mid \mathbf{x}_k)\, p(\mathbf{x}_k \mid \mathbf{y}_{1:k-1})\, \mathrm{d}\mathbf{x}_k. \tag{5.7}$$

The marginal measurement likelihood, which gives the predictive distribution of measurement $\mathbf{y}_k$ given all previous measurements, is given as

$$p(\mathbf{y}_k \mid \mathbf{y}_{1:k-1}) = \int p(\mathbf{y}_k \mid \mathbf{x}_k)\, p(\mathbf{x}_k \mid \mathbf{y}_{1:k-1})\, \mathrm{d}\mathbf{x}_k. \tag{5.8}$$

Note that this likelihood term is the same as the normalization constant Z_k given by Equation (5.7).

5.2 Kalman Filter

Kalman filter (KF) (see, e.g., Jazwinski, 1970; Maybeck, 1979; Bar-Shalom et al., 2001; Grewal and Andrews, 2001), which originally appeared in (Kalman, 1960), considers a dynamic linear model model

$$\mathbf{x}_k = \mathbf{A}_{k-1}\mathbf{x}_{k-1} + \mathbf{q}_{k-1} \tag{5.9}$$

$$\mathbf{y}_k = \mathbf{H}_k\mathbf{x}_k + \mathbf{r}_k, \tag{5.10}$$

where

$$\mathbf{q}_{k-1} \sim N(\mathbf{0}, \mathbf{Q}_{k-1}) \tag{5.11}$$

$$\mathbf{r}_k \sim N(\mathbf{0}, \mathbf{R}_k). \tag{5.12}$$

The symbols are defined as follows:

- $\mathbf{x}_k = \mathbf{x}(t_k)$ is the hidden state at time step t_k. This state can include, for example, the velocity and position of a vehicle $\mathbf{x} = (x\ y\ \dot{x}\ \dot{y})^T$.
- $\mathbf{y}_k$ is the implicit measurement that can be made from state $\mathbf{x}$ at time t_k. This measurement can be for example noisy position estimate given by some sensor or direction/distance/velocity measurement of radar or anything else.

- $\mathbf{A}_{k-1}$ is the transition matrix of dynamic behavior which defines how state $\mathbf{x}_{k-1}$ evolves on average when time passes from t_{k-1} to t_k. This matrix can result from integrating continuous time model from last measurement time instance at t_{k-1} to current at t_k using the methods presented in Chapter 3. In that case $\mathbf{A}_{k-1}$ is exactly the transition matrix of the continuous solution.

- $\mathbf{Q}_{k-1}$ defines the increase of uncertainty in the discrete dynamic model during the transition from t_{k-1} to t_k. This uncertainty can be integrated from the continuous model using the methods for noisy differential equations as described in Chapter 3.

- $\mathbf{H}_k$ defines the measurement process.

- $\mathbf{R}_k$ is the covariance of noise in the measurement procedure.

We have presented the model without input signal, since it is irrelevant in our application. Matrices $\mathbf{A}_{k-1}$, $\mathbf{H}_k$, $\mathbf{Q}_{k-1}$ and $\mathbf{R}_k$ are assumed known for all $k > 0$. The initial configuration defined by parameters $\mathbf{m}_0$ and $\mathbf{P}_0$ is also assumed to be known.

In probabilistic terms, the model is

$$p(\mathbf{y}_k \mid \mathbf{x}_k) = N(\mathbf{y}_k \mid \mathbf{H}_k\mathbf{x}_k, \mathbf{R}_k) \tag{5.13}$$

$$p(\mathbf{x}_k \mid \mathbf{x}_{k-1}) = N(\mathbf{x}_k \mid \mathbf{A}_{k-1}\mathbf{x}_{k-1}, \mathbf{Q}_{k-1}) \tag{5.14}$$

$$p(\mathbf{x}_0) = N(\mathbf{x}_0 \mid \mathbf{m}_0, \mathbf{P}_0). \tag{5.15}$$

Kalman Filter prediction step is described as Algorithm 3 and update step as Algorithm 4. The marginal measurement likelihood calculation (Equation (5.8)) for Kalman filter is given as Algorithm 5.

Synopsis:

$$[\mathbf{m}_k^-, \mathbf{P}_k^-] = \mathrm{KF}_p(\mathbf{m}_{k-1}, \mathbf{P}_{k-1}, \mathbf{A}_{k-1}, \mathbf{Q}_{k-1})$$

Data : Previous state mean $\mathbf{m}_{k-1}$ and state covariance $\mathbf{P}_{k-1}$. Model parameters: transition matrix $\mathbf{A}_{k-1}$ and process covariance $\mathbf{Q}_{k-1}$.

Result : Calculates predicted state mean $\mathbf{m}_k^-$ and covariance $\mathbf{P}_k^-$.

Calculate predicted mean and covariance:

$$\mathbf{m}_k^- = \mathbf{A}_{k-1}\mathbf{m}_{k-1} \tag{5.16}$$

$$\mathbf{P}_k^- = \mathbf{A}_{k-1}\mathbf{P}_{k-1}\mathbf{A}_{k-1}^T + \mathbf{Q}_{k-1} \tag{5.17}$$

Algorithm 3: Kalman Filter prediction step

Synopsis:

$$[\mathbf{m}_k, \mathbf{P}_k, \mathbf{v}_k, \mathbf{S}_k, \mathbf{K}_k] = \mathrm{KF}_u(\mathbf{y}_k, \mathbf{m}_k^-, \mathbf{P}_k^-, \mathbf{H}_k, \mathbf{R}_k)$$

Data : Measurement $\mathbf{y}_k$, predicted state mean $\mathbf{m}_k^-$ and state covariance $\mathbf{P}_k^-$. Model parameters: measurement matrix $\mathbf{H}_k$ and measurement covariance $\mathbf{R}_k$.

Result : Calculates updated state mean $\mathbf{m}_k$, state covariance $\mathbf{P}_k$, innovation mean $\mathbf{v}_k$, innovation covariance $\mathbf{S}_k$ and Kalman gain $\mathbf{K}_k$,

Calculate innovation mean and covariance:

$$\mathbf{v}_k = \mathbf{y}_k - \mathbf{H}_k \mathbf{m}_k^- \tag{5.18}$$

$$\mathbf{S}_k = \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{R}_k \tag{5.19}$$

Calculate gain, updated state mean and covariance:

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}_k^T \mathbf{S}_k^{-1} \tag{5.20}$$

$$\mathbf{m}_k = \mathbf{m}_k^- + \mathbf{K}_k \mathbf{v}_k \tag{5.21}$$

$$\mathbf{P}_k = \mathbf{P}_k^- - \mathbf{K}_k \mathbf{S}_k \mathbf{K}_k^T \tag{5.22}$$

Algorithm 4: Kalman Filter update step

Synopsis:

$$\rho = \mathrm{KF}_{lh}(\mathbf{y}_k, \mathbf{m}_k^-, \mathbf{P}_k^-, \mathbf{H}_k, \mathbf{R}_k)$$

Data : Measurement $\mathbf{y}_k$, predicted state mean $\mathbf{m}_k^-$ and state covariance $\mathbf{P}_k^-$. Model parameters: measurement matrix $\mathbf{H}_k$ and measurement covariance $\mathbf{R}_k$.

Result : Calculates the marginal measurement likelihood $p(\mathbf{y}_k \mid \mathbf{y}_{1:k-1})$.

Calculate innovation mean and covariance:

$$\mathbf{v}_k = \mathbf{y}_k - \mathbf{H}_k \mathbf{m}_k^- \tag{5.23}$$

$$\mathbf{S}_k = \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{R}_k \tag{5.24}$$

Calculate the probability:

$$\rho = N(\mathbf{v}_k \mid \mathbf{0}, \mathbf{S}_k) \tag{5.25}$$

Algorithm 5: Kalman Filter marginal measurement likelihood

In probabilistic terms, the Kalman Filter equations can be used for calculating parameters of the following distributions:

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:k-1}) = N(\mathbf{x}_k \mid \mathbf{m}_k^-, \mathbf{P}_k^-) \tag{5.26}$$

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:k}) = N(\mathbf{x}_k \mid \mathbf{m}_k, \mathbf{P}_k) \tag{5.27}$$

$$p(\mathbf{r}_k \mid \mathbf{y}_{1:k-1}) = N(\mathbf{r}_k \mid \mathbf{v}_k, \mathbf{S}_k), \tag{5.28}$$

where residual or innovation sequence is defined as

$$\mathbf{r}_k = \mathbf{y}_k - \mathbf{H}_k \mathbf{x}_k. \tag{5.29}$$

The original derivation of Kalman filter (Kalman, 1960) was based on computing sequential orthogonal projections of new measurements $\mathbf{y}_k$ into linear space of the measurements from the previous steps $\mathcal{Y}_{k-1}$. The basis of this derivation is in linear Gaussian processes even thought the derivation is based on optimality property of orthogonal projection in linear spaces. Alternative derivation, based on probabilistic formulation is given in Appendix A.1.

5.3 Extended Kalman Filter

Extended Kalman Filter (EKF) (see, e.g., Jazwinski, 1970; Maybeck, 1982a; Bar-Shalom et al., 2001; Grewal and Andrews, 2001) is a nonlinear extension of Kalman Filter. The model is (without input for simplicity):

$$\begin{aligned} \mathbf{x}_k &= \mathbf{a}(\mathbf{x}_{k-1}, \mathbf{q}_{k-1}) & \mathbf{q}_{k-1} &\sim N(0, \mathbf{Q}_{k-1}) \\ \mathbf{y}_k &= \mathbf{h}(\mathbf{x}_k, \mathbf{r}_k), & \mathbf{r}_k &\sim N(0, \mathbf{R}_k). \end{aligned} \tag{5.30}$$

The state mean and observation mean are approximated by nonlinearities evaluated at mean values of noise terms

$$E[\mathbf{x}_k] \approx \mathbf{a}(\mathbf{m}_{k-1}, 0) = \mathbf{m}_k^- \tag{5.31}$$

$$E[\mathbf{y}_k] \approx \mathbf{h}(\mathbf{m}_k^-, 0) = \bar{\mathbf{y}}_k. \tag{5.32}$$

In order to estimate covariances, the following linearization is made

$$\mathbf{x}_k \approx \mathbf{m}_k^- + \mathbf{A}_{k-1}(\mathbf{x}_{k-1} - \mathbf{m}_{k-1}) + \mathbf{W}_{k-1}\mathbf{q}_{k-1} \tag{5.33}$$

$$\mathbf{y}_k \approx \bar{\mathbf{y}}_k + \mathbf{H}_k(\mathbf{x}_{k-1} - \mathbf{m}_k^-) + \mathbf{V}_k\mathbf{r}_k, \tag{5.34}$$

where

$$\mathbf{A}_{k-1} = \left.\frac{\partial \mathbf{a}(\mathbf{x}, \mathbf{q})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{m}_{k-1}, \mathbf{q}=\mathbf{0}} \tag{5.35}$$

$$\mathbf{W}_{k-1} = \left.\frac{\partial \mathbf{a}(\mathbf{x}, \mathbf{q})}{\partial \mathbf{q}}\right|_{\mathbf{x}=\mathbf{m}_{k-1}, \mathbf{q}=\mathbf{0}} \tag{5.36}$$

$$\mathbf{H}_k = \left.\frac{\partial \mathbf{h}(\mathbf{x}, \mathbf{r})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{m}_k^-, \mathbf{r}=\mathbf{0}} \tag{5.37}$$

$$\mathbf{V}_k = \left.\frac{\partial \mathbf{h}(\mathbf{x}, \mathbf{r})}{\partial \mathbf{r}}\right|_{\mathbf{x}=\mathbf{m}_k^-, \mathbf{r}=\mathbf{0}}. \tag{5.38}$$

With these approximations the Extended Kalman Filter prediction equations take the form of Algorithm 6 and update equations form given of Algorithm 7.

Synopsis:

$$[\mathbf{m}_k^-, \mathbf{P}_k^-] = \mathrm{EKF}_p(\mathbf{m}_{k-1}, \mathbf{P}_{k-1}, \mathbf{a}(\mathbf{x}, \mathbf{q}), \mathbf{A}(\mathbf{x}, \mathbf{q}), \mathbf{Q}_{k-1}, \mathbf{W}(\mathbf{x}, \mathbf{q}))$$

Data : Previous state mean $\mathbf{m}_{k-1}$ and state covariance $\mathbf{P}_{k-1}$. Parameter $\mathbf{a}(\mathbf{x}, \mathbf{q})$ defines the dynamic transition function. $\mathbf{A}(\mathbf{x}, \mathbf{q})$ is derivative function of **a** with respect to state, $\mathbf{Q}_{k-1}$ is the process covariance and $\mathbf{W}(\mathbf{x}, \mathbf{q})$ is the derivative function of **a** with respect to noise.

Result : Calculates predicted state mean $\mathbf{m}_k^-$ and covariance $\mathbf{P}_k^-$.

Calculate mean prediction:

$$\mathbf{m}_k^- = \mathbf{a}(\mathbf{m}_{k-1}, 0) \tag{5.39}$$

Calculate predicted covariance:

$$\mathbf{A}_{k-1} = \mathbf{A}(\mathbf{m}_{k-1}, \mathbf{0}) \tag{5.40}$$

$$\mathbf{W}_{k-1} = \mathbf{W}(\mathbf{m}_{k-1}, \mathbf{0}) \tag{5.41}$$

$$\mathbf{P}_k^- = \mathbf{W}_{k-1}\mathbf{Q}_{k-1}\mathbf{W}_{k-1}^T + \mathbf{A}_{k-1}\mathbf{P}_{k-1}\mathbf{A}_{k-1}^T \tag{5.42}$$

Algorithm 6: Extended Kalman Filter prediction step

Synopsis:

$$[\mathbf{m}_k, \mathbf{P}_k, \mathbf{v}_k, \mathbf{S}_k, \mathbf{K}_k] = \mathrm{EKF}_u(\mathbf{y}_k, \mathbf{m}_k^-, \mathbf{P}_k^-, \mathbf{h}(\mathbf{x}, \mathbf{r}), \mathbf{H}(\mathbf{x}, \mathbf{r}), \mathbf{R}_k, \mathbf{V}(\mathbf{x}, \mathbf{r}))$$

Data : Measurement $\mathbf{y}_k$, predicted state mean $\mathbf{m}_k^-$ and state covariance $\mathbf{P}_k^-$. Parameter $\mathbf{h}(\mathbf{x}, \mathbf{r})$ defines measurement function. $\mathbf{H}(\mathbf{x}, \mathbf{r})$ is derivative function of $\mathbf{h}$ with respect to state. $\mathbf{R}_k$ is the measurement covariance and $\mathbf{V}(\mathbf{x}, \mathbf{r})$ is derivative function of $\mathbf{h}$ with respect to noise.

Result : Calculates updated state mean $\mathbf{m}_k$, state covariance $\mathbf{P}_k$, innovation mean $\mathbf{v}_k$, innovation covariance $\mathbf{S}_k$ and Kalman gain $\mathbf{K}_k$,

Evaluate the derivatives:

$$\mathbf{H}_k = \mathbf{H}(\mathbf{m}_k^-, \mathbf{0}) \tag{5.43}$$

$$\mathbf{V}_k = \mathbf{V}(\mathbf{m}_k^-, \mathbf{0}) \tag{5.44}$$

Calculate innovation mean and covariance:

$$\mathbf{v}_k = \mathbf{y}_k - \mathbf{h}(\mathbf{m}_k^-, \mathbf{0}) \tag{5.45}$$

$$\mathbf{S}_k = \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{V}_k \mathbf{R}_k \mathbf{V}_k^T \tag{5.46}$$

Calculate gain, updated state mean and covariance:

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}_k^T \mathbf{S}_k^{-1} \tag{5.47}$$

$$\mathbf{m}_k = \mathbf{m}_k^- + \mathbf{K}_k \mathbf{v}_k \tag{5.48}$$

$$\mathbf{P}_k = \mathbf{P}_k^- - \mathbf{K}_k \mathbf{S}_k \mathbf{K}_k^T \tag{5.49}$$

Algorithm 7: Extended Kalman Filter update step

Synopsis:

$$\rho = \mathrm{EKF}_{lh}(\mathbf{y}_k, \mathbf{m}_k^-, \mathbf{P}_k^-, \mathbf{h}(\mathbf{x},\mathbf{r}), \mathbf{H}(\mathbf{x},\mathbf{r}), \mathbf{R}_k, \mathbf{V}(\mathbf{x},\mathbf{r}))$$

Data : Measurement $\mathbf{y}_k$, predicted state mean $\mathbf{m}_k^-$ and state covariance $\mathbf{P}_k^-$. Parameter $\mathbf{h}(\mathbf{x},\mathbf{r})$ defines measurement function. $\mathbf{H}(\mathbf{x},\mathbf{r})$ is derivative function of $\mathbf{h}$ with respect to state. $\mathbf{R}_k$ is the measurement covariance and $\mathbf{V}(\mathbf{x},\mathbf{r})$ is derivative function of $\mathbf{h}$ with respect to noise.

Result : Calculates the marginal measurement likelihood $p(\mathbf{y}_k \mid \mathbf{y}_{1:k-1})$.

Evaluate the derivatives:

$$\mathbf{H}_k = \mathbf{H}(\mathbf{m}_k^-, \mathbf{0}) \qquad (5.50)$$
$$\mathbf{V}_k = \mathbf{V}(\mathbf{m}_k^-, \mathbf{0}) \qquad (5.51)$$

Calculate innovation mean and covariance:

$$\mathbf{v}_k = \mathbf{y}_k - \mathbf{h}(\mathbf{m}_k^-, \mathbf{0}) \qquad (5.52)$$
$$\mathbf{S}_k = \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{V}_k \mathbf{R}_k \mathbf{V}_k^T \qquad (5.53)$$

Calculate the probability:

$$\rho = N(\mathbf{v}_k \mid \mathbf{0}, \mathbf{S}_k) \qquad (5.54)$$

Algorithm 8: Extended Kalman Filter marginal measurement likelihood

5.4 Unscented Kalman Filter

Unscented Kalman Filter (UKF) (Julier et al., 1995; Wan and van der Merwe, 2001) is based on Unscented Transformation (UT) (Julier and Uhlmann, 1995), which is method for generating approximations of Gaussian distributions when they are propagated through nonlinear functions. Classical way of establishing this approximation is linearization as in case of EKF, but Unscented Transformation provides an alternative.

Assume that variable $\mathbf{x} \in \mathbb{R}^n$ has Gaussian distribution $N(\mathbf{m}_x, \mathbf{P}_{xx})$ and this variable is propagated through nonlinear function $\mathbf{y} = g(\mathbf{x})$, and the purpose is to generate the best possible Gaussian approximation for distribution of variable $\mathbf{y}$. The classical *linear approximation* already employed in EKF would be

$$\mathbf{m}_y \approx g(\mathbf{m}_x) \tag{5.55}$$
$$\mathbf{P}_{\mathbf{yy}} \approx \mathbf{G}\,\mathbf{P}_{xx}\,\mathbf{G}^T, \tag{5.56}$$

where $\mathbf{G}$ is the Jacobian of $g(\mathbf{x})$ evaluated at $x = \mathbf{m}_x$

$$\mathbf{G} = \left.\frac{\mathrm{d}g(\mathbf{x})}{\mathrm{d}\mathbf{x}}\right|_{x=\mathbf{m}_x}. \tag{5.57}$$

The unscented transformation (Julier and Uhlmann, 1995) works very differently – the idea is to generate fixed number of *sigma points*, which capture the mean and covariance of original distribution exactly. Those sigma points are propagated one at a time through function $g(\cdot)$ and new transformed distribution is estimated from them. The sigma points are selected so that distribution of $\mathbf{y}$ can be estimated from these transformed sigma points as accurately as possible. The procedure is presented as Algorithm 9.

In Algorithm 9, the matrix square root of positive definite matrix $\mathbf{P}$ means a matrix $\mathbf{S} = \sqrt{\mathbf{P}}$ such that

$$\mathbf{P} = \mathbf{S}\,\mathbf{S}^T. \tag{5.69}$$

Since the only requirement to $\mathbf{S}$ is the definition above, we can for example use the lower triangular matrix of *Cholesky factorization* (see, e.g., Kreyszig, 1993).

Unscented Kalman Filter is an approximate solution to the same nonlinear state model as EKF, but instead of linearization, it uses Unscented Transformation. The state model is

$$\begin{aligned} \mathbf{x}_k &= \mathbf{a}(\mathbf{x}_{k-1}, \mathbf{w}_{k-1}) & \mathbf{w}_{k-1} &\sim N(0, \mathbf{Q}_{k-1}) \\ \mathbf{y}_k &= \mathbf{h}(\mathbf{x}_k, \mathbf{v}_k), & \mathbf{v}_k &\sim N(0, \mathbf{R}_k). \end{aligned} \tag{5.70}$$

The UKF computations are described as Algorithm 10. If the process noise model is additive (as usually is the case) the equations in Algorithm 10 can be simplified (see Wan and van der Merwe, 2001).

5.5 Sequential Importance Resampling

Sequential Importance Sampling (SIR) (see, e.g., Doucet et al., 2001) can be used for estimating recursively the moments of model

$$\mathbf{x}_k \sim p(\mathbf{x}_k \mid \mathbf{x}_{k-1}) \tag{5.84}$$
$$\mathbf{y}_k \sim p(\mathbf{y}_k \mid \mathbf{x}_k), \tag{5.85}$$

1. Compute the set of $2n+1$ points from the rows or columns of the matrix $\sqrt{(n+\lambda)\mathbf{P}_{xx}}$:

$$\mathbf{X}_0 = \mathbf{m}_x \tag{5.58}$$

$$\mathbf{X}_i = \mathbf{m}_x + \left(\sqrt{(n+\lambda)\mathbf{P}_{xx}}\right)_i, \quad i = 1,\ldots,n \tag{5.59}$$

$$\mathbf{X}_i = \mathbf{m}_x - \left(\sqrt{(n+\lambda)\mathbf{P}_{xx}}\right)_i, \quad i = n+1,\ldots,2n \tag{5.60}$$

and the associated weights:

$$W_0^{(m)} = \lambda/(n+\lambda) \tag{5.61}$$

$$W_0^{(c)} = \lambda/(n+\lambda) + (1-\alpha^2+\beta) \tag{5.62}$$

$$W_i^{(m)} = 1/\{2(n+\lambda)\}, \quad i = 1,\ldots,2n \tag{5.63}$$

$$W_i^{(c)} = 1/\{2(n+\lambda)\}, \quad i = 1,\ldots,2n \tag{5.64}$$

Parameter λ is a scaling parameter defined as

$$\lambda = \alpha^2(n+\kappa) - n \tag{5.65}$$

The positive constants α, β and κ are are used as parameters of the method.

2. Transform each of the sigma points as

$$\mathbf{Y}_i = g(\mathbf{X}_i), \quad i = 0,\ldots,2n \tag{5.66}$$

3. Mean and covariance estimates for $\mathbf{y}$ can be calculated as

$$\mathbf{m}_y \approx \sum_{i=0}^{2n} W_i^{(m)} \mathbf{Y}_i \tag{5.67}$$

$$\mathbf{P}_{yy} \approx \sum_{i=0}^{2n} W_i^{(c)} (\mathbf{Y}_i - \mathbf{m}_y)(\mathbf{Y}_i - \mathbf{m}_y)^T \tag{5.68}$$

Algorithm 9: Unscented Transformation

Initialize: Calculate the augmented prior mean and covariance:

$$\mathbf{m}_0^a = E[\mathbf{x}_0^a] = \left(\mathbf{m}_0^T \ \mathbf{0} \ \mathbf{0}\right)^T \tag{5.71}$$

$$\mathbf{P}_0^a = E[(\mathbf{x}_0^a - \mathbf{m}_0^a)(\mathbf{x}_0^a - \mathbf{m}_0^a)^T] = \begin{pmatrix} \mathbf{P}_0 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_0 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{R}_1 \end{pmatrix} \tag{5.72}$$

Sigma points: Use the unscented transformation method for generating sigma points:

$$\mathbf{X}_{k-1} = \left[\mathbf{m}_{k-1}^a \ \mathbf{m}_{k-1}^a \pm \left(\sqrt{(n+\lambda)\mathbf{P}_{k-1}^a}\right)_i\right] \tag{5.73}$$

Prediction: Estimate predicted state distribution from the sigma points:

$$\mathbf{X}_{k|k-1} = \mathbf{a}(\mathbf{X}_{k-1}^x, \mathbf{X}_{k-1}^w) \tag{5.74}$$

$$\mathbf{m}_k^- = \sum_{i=0}^{2n} W_i^{(m)} \mathbf{X}_{i,k|k-1} \tag{5.75}$$

$$\mathbf{P}_k^- = \sum_{i=0}^{2n} W_i^{(c)} (\mathbf{X}_{i,k|k-1} - \mathbf{m}_k^-)(\mathbf{X}_{i,k|k-1} - \mathbf{m}_k^-)^T \tag{5.76}$$

Update: Estimate measurement covariance and posterior state distribution from the sigma points:

$$\mathbf{Y}_{k|k-1} = \mathbf{h}(\mathbf{X}_{k|k-1}^x, \mathbf{X}_{k-1}^v) \tag{5.77}$$

$$\overline{\mathbf{y}}_k^- = \sum_{i=0}^{2n} W_i^{(m)} \mathbf{Y}_{i,k|k-1} \tag{5.78}$$

$$\mathbf{P}_{y_k,y_k} = \sum_{i=0}^{2n} W_i^{(c)} (\mathbf{Y}_{i,k|k-1} - \overline{\mathbf{y}}_k^-)(\mathbf{Y}_{i,k|k-1} - \overline{\mathbf{y}}_k^-)^T \tag{5.79}$$

$$\mathbf{P}_{x_k,y_k} = \sum_{i=0}^{2n} W_i^{(c)} (\mathbf{X}_{i,k|k-1} - \mathbf{m}_k^-)(\mathbf{Y}_{i,k|k-1} - \overline{\mathbf{y}}_k^-)^T \tag{5.80}$$

$$\mathbf{K}_k = \mathbf{P}_{x_k,y_k} \mathbf{P}_{y_k,y_k}^{-1} \tag{5.81}$$

$$\mathbf{m}_k = \mathbf{m}_k^- + \mathbf{K}_k (\mathbf{y}_k - \overline{\mathbf{y}}_k^-) \tag{5.82}$$

$$\mathbf{P}_k = \mathbf{P}_k^- - \mathbf{K}_k \mathbf{P}_{y_k,y_k} \mathbf{K}_k^T \tag{5.83}$$

Algorithm 10: Unscented Kalman Filter

where $\mathbf{x}_k$ is the state, and $\mathbf{y}_k$ is the measurement as in case of generic Bayesian Filtering. In addition to model above there is importance distribution $\pi(\cdot)$, which is an approximation to posterior distribution of states given the value of previous step

$$\pi(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{y}_{1:k}) \approx p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{y}_{1:k}). \tag{5.86}$$

The importance distribution should be in such functional form that we can draw samples from that distribution and evaluate probability densities of those points.

Algorithm is started from initial distribution, whic is given in form of prior sample set $\{\mathbf{x}_0^{(i)}, i = 1, \ldots, N\}$ and corresponding weights $w_0^{(i)}$. Then the SIR step described as Algorithm 11 is repeated for each measurements. SIR uses resampling Algorithm 12 sub-program. This is the basic resampling method and more advanced resampling methods can be found, for example, in article (Kitagawa, 1996).

On every step, the expectation of any function $\mathbf{g}(\mathbf{x})$ in posterior distribution of $\mathbf{x}_k$ can be calculated as weighted sample average

$$E[\mathbf{g}(\mathbf{x}_k)] \approx \sum_{i=1}^{N} w_k^{(i)} \mathbf{g}(\mathbf{x}_k^{(i)}). \tag{5.93}$$

Bootstrap filter (Gordon et al., 1993) is a special case of Sequential Importance Resampling, where importance distribution is $\pi(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{y}_{1:k}) = p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$ and resampling is performed on every step. Optimal importance distribution is given as $\pi(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{y}_{1:k}) = p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{y}_{1:k})$, which can be used in practice for example in Monte Carlo data association case.

In SIR resampling is not performed on every step, but only when it is actually needed. One way of implementing this is to do resampling on every k'th step, where k is some predefined constant. This method has the advantage that it is unbiased. Another way, which is also used in our simulation system, is *adaptive resampling*, where effective number of weights is used for monitoring the need for resampling. Adaptive resampling has the disadvantage that in theory, additional adaptive term in algorithm may introduce bias in estimation result. However, this bias is negligible in most cases.

An estimate for effective number of particles based on an approximation of the variance of importance weights can be computed as:

$$n_{\text{eff}} \approx \frac{1}{\sum_{i=1}^{N} \left(w_k^{(i)}\right)^2}, \tag{5.94}$$

where $w_k^{(i)}$ is the normalized weight of particle i on time step k (Liu and Chen, 1995). Resampling is performed when effective number of particles is significantly less than total number of particles. In our simulation system, we resample if $n_{\text{eff}} < N/4$, where N is the total number of particles.

```
Synopsis:

    $[\mathcal{X}_k, \mathcal{W}_k] = \mathrm{SIR}(\mathcal{X}_{k-1}, \mathcal{W}_{k-1}, p(\mathbf{x}_k \mid \mathbf{x}_{k-1}), p(\mathbf{y}_k \mid \mathbf{x}_k), \pi(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{y}_{1:k}))$

Data : Previous state distribution as weighted sample set
       $(\mathcal{X}_{k-1}, \mathcal{W}_{k-1}) = \{\mathbf{x}_{k-1}^{(i)}, w_{k-1}^{(i)} : i = 1, \ldots, N\}$ Model parameters: transition distribution $p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$, measurement
       likelihood distribution $p(\mathbf{y}_k \mid \mathbf{x}_k)$ and importance distribution
       $\pi(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{y}_{1:k})$.
Result : Predicted and updated state distribution as weighted sample set
       $(\mathcal{X}_k, \mathcal{W}_k) = \{\mathbf{x}_k^{(i)}, w_k^{(i)} : i = 1, \ldots, N\}$
Draw new point $\mathbf{x}_k^{(i)}$ for each point in sample set $\{\mathbf{x}_{k-1}^{(i)}, i = 1, \ldots, N\}$
from the importance distribution:

    $\mathbf{x}_k^{(i)} \sim \pi(\mathbf{x}_k \mid \mathbf{x}_{k-1}^{(i)}, \mathbf{y}_{1:k})$    (5.87)

Calculate new weights as follows:

    $w_k^{(i)} \propto w_{k-1}^{(i)} \frac{p(\mathbf{y}_k \mid \mathbf{x}_k^{(i)})\, p(\mathbf{x}_k^{(i)} \mid \mathbf{x}_{k-1}^{(i)})}{\pi(\mathbf{x}_k^{(i)} \mid \mathbf{x}_{k-1}^{(i)}, \mathbf{y}_{1:k})}$    (5.88)

and normalize them sum to unity.
if (effective number of weights is too low) then
    $[\mathcal{X}_k, \mathcal{W}_k] = \mathrm{Resample}(\mathcal{X}_k, \mathcal{W}_k)$
end
```

Algorithm 11: Sequential Importance Resampling (SIR) prediction and update.

Good importance distributions can be obtained by *local linearization* where mixtures of Extended Kalman Filters (EKF) or Unscented Kalman Filters (UKF) are used as importance distribution (Doucet, 1998; van der Merwe et al., 2001). van der Merwe et al. (2001) also suggest a Metropolis-Hastings step after (or in place of) resampling step to smooth the resulting distribution, but from their results, it seems that this extra computation step has no significant performance effect. A particle filter with UKF importance distribution is often referred as *Unscented Particle Filter* (UPF).

```
Synopsis:
    $[\mathcal{X}'_k, \mathcal{W}'_k] = \mathrm{Resample}(\mathcal{X}_k, \mathcal{W}_k)$
Data   : Weighted sample set $(\mathcal{X}_k, \mathcal{W}_k) = \{\mathbf{x}_k^{(i)}, w_k^{(i)} : i = 1, \ldots, N\}$.
Result : Weighted sample set $(\mathcal{X}'_k, \mathcal{W}'_k) = \{\mathbf{x}_k^{'(i)}, w_k^{'(i)} : i = 1, \ldots, N\}$.
for $i = 1, \ldots, N$ do
    Draw uniform random number:
        $u \sim U(0, 1)$                                  (5.89)
    Find the smallest index $i'$ such that cumulative sum of normalized
    weights is greater or equal to $u$:
        $\sum_{j=1}^{i'} w_k^{(j)} \geq u$                 (5.90)
    and set
        $\mathbf{x}^{'(i)} = \mathbf{x}^{(i')}$            (5.91)
        $w_k^{'(i)} = 1/N$                                 (5.92)
end
```

Algorithm 12: Resampling for SIR algorithm

5.6 Rao-Blackwellized Particle Filter

The idea of Rao-Blackwellized particle filtering (see, e.g., Akashi and Kumamoto, 1977; Doucet, 1998; Chen and Liu, 2000; Doucet et al., 2001; Gustafsson et al., 2002) is that sometimes it is possible to calculate part of the filtering equations analytically and the other part by Monte Carlo sampling instead of calculating everything by pure sampling. According to Rao-Blackwell theorem this leads to estimators with much less variance than could be obtained by pure Monte Carlo sampling (Casella and Robert, 1996). An intuitive way of thinking this is that marginalization replaces the finite Monte Carlo particle set representation by an infinite closed form particle set, and that infinite set of particles is always more accurate than any finite set.

Consider the following Conditional Dynamic Linear Model (CDLM)

$$\mathbf{x}_k = \mathbf{A}(\lambda_{k-1})\mathbf{x}_{k-1} + \mathbf{q}_{k-1} \tag{5.95}$$

$$\mathbf{y}_k = \mathbf{H}(\lambda_k)\mathbf{x}_k + \mathbf{r}_k, \tag{5.96}$$

where

$$\mathbf{q}_{k-1} \sim N(\mathbf{0}, \mathbf{Q}(\lambda_{k-1})) \tag{5.97}$$

$$\mathbf{r}_k \sim N(\mathbf{0}, \mathbf{R}(\lambda_k)) \tag{5.98}$$

$$\lambda_k \sim p(\lambda_k \mid \lambda_{k-1}). \tag{5.99}$$

The model is conditionally linear, since given parameters λ, the model is Linear Gaussian. The idea of Rao-Blackwellization is that we can integrate Gaussian parts of model in closed form with Kalman Filter and use particle filter for remaining part of model.

5.7 Other Methods

- **Second Order Extended Kalman Filter** (Bar-Shalom et al., 2001) uses second order Taylor series expansion of the non-linearities instead of the first order linearization used in standard EKF.
- **Unscented Kalman Filter** (Julier et al., 1995; Wan and van der Merwe, 2001) is based on *Unscented Transformation*, which is used for generating Gaussian approximation of state distribution instead of EKF's linearization.
- **Non-Linear Projection Filter** (Gunther et al., 1997) is able to approximate non-linear dynamic models especially well. The idea is to integrate the Fokker-Planck-Kolmogorov equation of non-linear dynamic model by *method of Galerkin* (see, e.g., Guenther and Lee, 1988). Finite Element Method (FEM) is a special case of Galerkin method with localized basis functions.

Chapter 6

Approaches to Tracking Multiple Targets in Clutter

6.1 Multiple Hypothesis Tracking for Clutter

Clutter can be modeled such that, in addition to the actual measurement, we observe an indicator variable λ_k, which defines the association event:

$$\lambda_k = 0 : \{\text{measurement } k \text{ originated from false alarm}\} \tag{6.1}$$

$$\lambda_k = 1 : \{\text{measurement } k \text{ originated from target}\} \tag{6.2}$$

If we denote the whole history of measurement source indicators until the time step k with $\Lambda_k = \{\lambda_1, \ldots, \lambda_k\}$. On one step we might have multiple measurements but assuming that the measurements are conditionally independent we can treat all measurements sequentially.

The likelihood is now joint likelihood of measurements and associations:

$$p(\mathbf{y}_k, \lambda_k | \mathbf{x}_k) = \{\text{probability of observation } \mathbf{y}_k \text{ with origin } \lambda_k \text{ on state } \mathbf{x}_k\}. \tag{6.3}$$

This likelihood can be factorized as

$$p(\mathbf{y}_k, \lambda_k | \mathbf{x}_k) = p(\mathbf{y}_k | \lambda_k, \mathbf{x}_k)\ p(\lambda_k | \mathbf{x}_k), \tag{6.4}$$

and we can model the measurement likelihood $p(\mathbf{y}_k | \lambda_k, \mathbf{x}_k)$ and association likelihood $p(\lambda_k | \mathbf{x}_k)$ separately.

For example, we could have a Gaussian measurement model for target originated measurements and the probability of false alarms could be constant δ. Then, we have the following likelihood for a single measurement $\mathbf{y}_k$:

$$p(\mathbf{y}_k | \Lambda_k, \mathbf{x}_k) = p(\mathbf{y}_k | \lambda_k, \mathbf{x}_k) = \begin{cases} \delta, & \lambda_k = 0 \\ N(\mathbf{y}_k | \mathbf{h}(\mathbf{x}_k, \mathbf{v}_k), \mathbf{R}_k), & \lambda_k = 1. \end{cases} \tag{6.5}$$

Since the measurement likelihood depends only on the current associations, we can calculate the posterior distribution of states as follows:

$$p(\mathbf{x}_k | \Lambda_k, \mathbf{y}_{1:k}) \propto p(\mathbf{y}_k, \lambda_k | \mathbf{x}_k) \; p(\mathbf{x}_k | \Lambda_{k-1}, \mathbf{y}_{1:k-1}). \tag{6.6}$$

The joint probability of states and latent variables is then

$$p(\mathbf{x}_k, \Lambda_k | \mathbf{y}_{1:k}) = p(\mathbf{x}_k | \Lambda_k, \mathbf{y}_{1:k}) \; p(\Lambda_k | \mathbf{y}_{1:k}).. \tag{6.7}$$

The unknown terms $p(\mathbf{x}_k | \Lambda_{k-1}, \mathbf{y}_{1:k-1})$ in (6.6) and $p(\Lambda_k | \mathbf{y}_{1:k})$ in (6.7) can be calculated using the Multiple Hypothesis Tracking (MHT) recursions (Reid, 1979; Bar-Shalom and Li, 1995; Blackman and Popoli, 1999; Stone et al., 1999) which are presented as Algorithm 13.

Generic Multiple Hypothesis Tracking Recursions:

$$p(\mathbf{x}_k | \mathbf{y}_{1:k-1}, \Lambda_{k-1}) = \int p(\mathbf{x}_k | \mathbf{x}_{k-1}) \; p(\mathbf{x}_{k-1} | \mathbf{y}_{1:k-1}, \Lambda_{k-1}) \, d\mathbf{x}_{k-1} \tag{6.8}$$

$$p(\Lambda_k | \mathbf{y}_{1:k}) \propto p(\Lambda_{k-1} | \mathbf{y}_{1:k-1}) \int p(\mathbf{y}_k, \lambda_k | \mathbf{x}_k) \; p(\mathbf{x}_k | \mathbf{y}_{1:k-1}, \Lambda_{k-1}) \, d\mathbf{x}_k \tag{6.9}$$

Algorithm 13: Multiple Hypothesis Tracking

On every step of MHT, we have to calculate $p(\Lambda_k | \mathbf{y}_{1:k})$ for each possible history of latent variables. This number of histories grows exponentially by number of steps and makes direct application of MHT computationally infeasible. In theory, MHT recursion always produces optimal estimators with respect to model hypotheses.

6.2 Probabilistic Data Association

Probabilistic data association (PDA) filter (Bar-Shalom and Li, 1995; Blackman and Popoli, 1999) has the following assumptions:

- There is only one target of interest having linear dynamical and measurement models

$$\begin{aligned} \mathbf{x}_k &= \mathbf{A}_{k-1}\mathbf{x}_{k-1} + \mathbf{w}_{k-1}, & \mathbf{w}_{k-1} &\sim N(0, \mathbf{Q}_{k-1}) \\ \mathbf{y}_k &= \mathbf{H}_k\mathbf{x}_k + \mathbf{v}_k, & \mathbf{v}_k &\sim N(0, \mathbf{R}_k). \end{aligned}$$

- The track has been initialized.

- The past information about the target is summarized approximately by Gaussian distribution
$$p(\mathbf{x}_k|\mathbf{y}_{1:k-1}) = N(\mathbf{x}_k|\mathbf{m}_k^-, \mathbf{P}_k^-). \tag{6.10}$$
- At each time, a *validation gate* $\mathcal{V}_k$ is set up and only measurement inside this region are accepted. *Validated measurements* are those measurements lying inside the validation area. This validation gate is defined as
$$\begin{aligned}\mathcal{V}_k &= \{\mathbf{y} \,:\, [\mathbf{y} - \hat{\mathbf{y}}_k^-]^T \mathbf{S}_k^{-1} [\mathbf{y} - \hat{\mathbf{y}}_k^-] \le \gamma\} \\ &= \{\mathbf{y} \,:\, \mathbf{v}_k^T \mathbf{S}_k^{-1} \mathbf{v}_k \le \gamma\},\end{aligned} \tag{6.11}$$
with suitable threshold value γ (see Bar-Shalom and Li, 1995, for details).
- Among the possibly several validated measurements only one is target originated – if the target was detected and the corresponding measurement fell into the validation region.
- The remaining measurements are assumed to be due to false alarm or clutter and are modeled as IID (Independent Identically Distributed) with uniform spatial distribution.
- The target detections occur independently over time with known probability P_D.

PDA consists of the steps described as Algorithm 14. Association probabilities $\beta_{k,i}$ needed in the algorithm are given for *parametric PDA with Poisson clutter model* as follows:

$$\beta_{k,i} = \begin{cases} \frac{e_i}{b+\sum_{j=1}^{m_k} e_j}, & i = 1, \ldots, m_k \\ \frac{b}{b+\sum_{j=1}^{m_k} e_j}, & i = 0, \end{cases} \tag{6.22}$$

where

$$e_i = e^{-\frac{1}{2}\mathbf{v}_{k,i}^T \mathbf{S}_k^{-1} \mathbf{v}_{k,i}} \tag{6.23}$$
$$b = \left(\frac{2\pi}{\gamma}\right) \lambda V_k c_{n_y}^{-1} \frac{1 - P_D P_G}{P_D}, \tag{6.24}$$

and V_k is the volume of gate, n_y is the dimension of measurements $\mathbf{y}$, P_G is probability of gate, c_{n_y} is the volume of n_y dimensional unit hypersphere and m_k is the number of measurements at time step k. Poisson model for clutter assumes that number of false measurements in validation region is

$$p_{\mathrm{FA}}(m) = e^{-\lambda V} \frac{(\lambda V)^m}{m!}, \tag{6.25}$$

Initialization: Set mean and covariance to values $\mathbf{m}_0, \mathbf{P}_0$ such that the prior state distribution is $\mathbf{x}_0 \sim N(\mathbf{m}_0, \mathbf{P}_0)$.
Prediction:
The prediction step is the same as with standard Kalman Filter:

$$\begin{aligned} \mathbf{m}_k^- &= \mathbf{A}_{k-1}\mathbf{m}_{k-1} \\ \mathbf{P}_k^- &= \mathbf{Q}_{k-1} + \mathbf{A}_{k-1}\mathbf{P}_{k-1}\mathbf{A}_{k-1}^T \end{aligned} \tag{6.12}$$

Gating: Select set of *validated measurements*

$$\mathbf{Y}_k = \{\mathbf{y}_k^{(i)}\}, \qquad i = 1, \ldots, m_k \tag{6.13}$$

for time step t_k by accepting only those measurements that lie inside the gate (6.11). Predicted mean $\hat{y}_k^-$ and covariance for the "true" measurement are given as

$$\hat{\mathbf{y}}_k^- = \mathbf{H}_k\mathbf{m}_k^- \tag{6.14}$$

$$\mathbf{S}_k = \mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T + \mathbf{R}_k \tag{6.15}$$

Update:
The mean and covariance updates are defined as follows:

$$\mathbf{m}_k = \mathbf{m}_k^- + \mathbf{K}_k\mathbf{v}_k \tag{6.16}$$

$$\mathbf{P}_k = \beta_{k,0}\mathbf{P}_k^- + (1-\beta_{k,0})\mathbf{P}_k^c + \tilde{\mathbf{P}}_k \tag{6.17}$$

where the Kalman Gain $\mathbf{K}_k$, combined innovation $\mathbf{v}_k$, the covariance of the correct measurement $\mathbf{P}_k^c$ and spread of the innovations term $\tilde{\mathbf{P}}_k$ are given as:

$$\mathbf{K}_k = \mathbf{P}_k^-\mathbf{H}_k^T\mathbf{S}_k^{-1} \tag{6.18}$$

$$\mathbf{v}_k = \sum_{i=1}^{m_k} \beta_{k,i}\mathbf{v}_{k,i} \tag{6.19}$$

$$\mathbf{P}_k^c = \mathbf{P}_k^- - \mathbf{K}_k\mathbf{S}_k\mathbf{K}_k \tag{6.20}$$

$$\tilde{\mathbf{P}}_k = \mathbf{K}_k\left[\sum_{i=1}^{m_k} \beta_{k,i}\mathbf{v}_{k,i}\mathbf{v}_{k,i}^T - \mathbf{v}_{k,i}\mathbf{v}_{k,i}^T\right]\mathbf{K}_k^T \tag{6.21}$$

Association probabilities can be calculated from (6.22).

Algorithm 14: Probabilistic Data Association

where λ is the spatial density of false alarms. Alternative model is the *diffuse prior model* which is also called as *nonparametric PDA*

$$p_{\mathrm{FA}}(m) \propto 1, \tag{6.26}$$

which has the same equations as the Poisson model except λV_k in (6.24) is replaced with m_k

$$b = \left(\frac{2\pi}{\gamma}\right) m_k c_{n_y}^{-1} \frac{1 - P_D P_G}{P_D}. \tag{6.27}$$

6.3 Bootstrap Filter for Clutter

By augmenting the state with the same latent variable that was used in Section 6.1, a very simple Bootstrap filter can be derived. Predictive distribution for joint distribution of state and latent variable history is

$$\begin{aligned} p(\mathbf{x}_k, \Lambda_k | \mathbf{y}_{1:k-1}) &= p(\lambda_k | \mathbf{x}_k)\; p(\mathbf{x}_k, \Lambda_{k-1} | \mathbf{y}_{1:k-1}) \\ &= p(\lambda_k | \mathbf{x}_k) \int p(\mathbf{x}_k | \mathbf{x}_{k-1})\; p(\mathbf{x}_{k-1}, \Lambda_{k-1} | \mathbf{y}_{1:k-1})\, d\mathbf{x}_{k-1}, \end{aligned} \tag{6.28}$$

with the assumption, that associations represented by the latent variables are independent on each step.

The likelihood could be (for example) given as

$$p(\mathbf{y}_k | \mathbf{x}_k, \Lambda_k) = p(\mathbf{y}_k | \mathbf{x}_k, \lambda_k) = \begin{cases} \rho_0, & \lambda_k = 0 \\ N(\mathbf{y}_k | \mathbf{h}(\mathbf{x}_k, \mathbf{v}_k), \mathbf{R}_k) & \lambda_k = 1. \end{cases} \tag{6.29}$$

We could use some better model for false alarms, but it can be easily included afterwards. The likelihood is not really restricted to Gaussian but it is used here for simplicity. So, the bootstrap filter would be as described in Algorithm 15. Note that the algorithm directly uses dynamic model as its importance distribution, which makes it Bootstrap filter.

All the enhancing methods described in Chapter 5 can used here, but this simple Bootstrap version is presented here as an example how this can be done.

6.4 Multiple Hypothesis Tracking with Multiple Targets

Multiple target tracking is an extension to single target tracking in clutter and for this reason the algorithms are very similar. *Multiple Hypothesis Tracking* (MHT),

Initialization: Sample N samples from prior distributions

$$\mathbf{x}_0^{(i)} \sim p(\mathbf{x}_0) \qquad \lambda_0^{(i)} \sim p(\lambda_0) \tag{6.30}$$

Prediction: Sample N new samples from

$$\mathbf{x}_k^{(i)} \sim p(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)}) \tag{6.31}$$

and corresponding latent variables from

$$\lambda_k^{(i)} \sim p(\lambda_k | \mathbf{x}_k^{(i)}) \tag{6.32}$$

Update: Evaluate importance weights

$$w_k^{(i)} \propto w_{k-1}^{(i)} p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \lambda_k^{(i)}) \tag{6.33}$$

and normalize them.

Resampling: Use the resampling scheme as in Algorithm 11.
Set $k \leftarrow k+1$ and go to step 2.

Algorithm 15: Bootstrap Filter for Clutter

which was described in Section 6.1, can be extended to multiple targets by defining the latent variable λ_k as follows:

$$\begin{aligned}
\lambda &= 0 : \{\text{measurement originated from false alarm}\} \\
\lambda &= 1 : \{\text{measurement originated from target 1}\} \\
\lambda &= 2 : \{\text{measurement originated from target 2}\} \\
&\vdots \\
\lambda &= T : \{\text{measurement originated from target } T\}.
\end{aligned} \tag{6.34}$$

With this definition the MHT recursions in Equations (6.8) and (6.9) can be used as such, but the complexity will be even more enormous. However, this kind of latent variable formulation can be used as starting point for developing more practical algorithms. The formulation doesn't take split measurements (one measurement that originated from multiple targets) into account, but it can be extended to that also. We may also obtain several measurements from each sensor during a measurement scan.

Bar-Shalom and Li (1995) note that since MHT approach is measurement oriented (opposite to target oriented), track initialization can be combined into the algorithm. With suitable definitions of latent variable states (e.g., new target state) it is possible to develop algorithms for tracking unknown number of targets. See the citation for details.

6.5 Joint Probabilistic Data Association Filter

Joint Probabilistic Data Association (JPDA) filter (see e.g., Bar-Shalom and Li, 1995; Blackman and Popoli, 1999, for more complete presentation) is an extension of Probabilistic Data Association (PDA) filter for multiple targets. The assumptions are the following:

- There is a known number of targets in clutter.
- The track of each target has been initialized
- Measurements from one target can fall in the validation region of a neighboring target – this can happen over several sampling times and acts as a persistent inference.
- A target can give a rise to at most one measurement.
- The detection of target occurs independently over time and from other targets according to a known probability.
- A measurement could have originated from at most one target.
- The past is summarized by an approximate sufficient statistic – state estimates (approximate conditional mean) and covariances for each target.
- The states are assumed Gaussian distributed with the above means and covariances.
- Each target has linear Gaussian dynamic and measurement models as in (6.10). The models for various targets don't have to be the same.

Validation matrix is defined as

$$\Omega = [\omega_{j,t}], \quad j = 1, \dots, m, \quad t = 0, 1, \dots, N_T, \tag{6.35}$$

with binary elements that indicate if measurement j lies in the validation region of target t. The index $t = 0$ stands for "none of the targets".

A joint association event θ is represented by the *event matrix*

$$\hat{\Omega} = [\hat{\omega}_{j,t}(\theta)], \tag{6.36}$$

consisting of the units in Ω corresponding to the associations in θ:

$$\hat{\omega}_{j,t} = \begin{cases} 1 & \text{if } \theta_{j,t} \in \theta \\ 0 & \text{otherwise.} \end{cases} \tag{6.37}$$

Target detection indicator is defined as

$$\delta_t(\theta) = \sum_{j=1}^{m} \hat{\omega}_{j,t}(\theta) \tag{6.38}$$

and *measurement association indicator* is defined as

$$\tau_j(\theta) = \sum_{t=1}^{N_T} \hat{\omega}_{j,t}(\theta) \tag{6.39}$$

The number of false measurements with these definitions is

$$\phi(\theta) = \sum_{j=1}^{m} (1 - \tau_j(\theta)). \tag{6.40}$$

The Algorithm of JPDA is shown in Algorithm 16. The state estimation equations for JPDA are the same as for PDA except that association probabilities $\beta_{k,i}$ are evaluated in different manner. Probabilities are given as

$$\begin{aligned} \beta_{k,j,t} &= p(\theta_{j,t} | \mathbf{Y}_k) \\ &= \sum_{\theta} p(\theta | \mathbf{Y}_k) \hat{\omega}_{j,t}(\theta). \end{aligned} \tag{6.50}$$

- *Decoupled parametric JPDA with Poisson clutter model* can be written as

$$p(\theta_k | \mathbf{Y}_k) \propto \prod_j \left\{ \lambda^{-1} f_{t_j}(\mathbf{y}_{k,j}) \right\}^{\tau_j} \prod_t \left(P_{t,D} \right)^{\delta_t} \left(1 - P_{t,D} \right)^{1-\delta_t}, \tag{6.51}$$

  where

$$f_{t_j}(\mathbf{y}_{k,j}) = N(\mathbf{y}_{k,j} | \hat{\mathbf{y}}^{-}_{k,t_j}, \mathbf{S}_{k,t_j}). \tag{6.52}$$

- *Decoupled nonparametric JPDA* can be written as

$$p(\theta_k | \mathbf{Y}_k) \propto \phi! \prod_j \left\{ V f_{t_j}(\mathbf{y}_{k,j}) \right\}^{\tau_j} \prod_t \left(P_{t,D} \right)^{\delta_t} \left(1 - P_{t,D} \right)^{1-\delta_t}, \tag{6.53}$$

  with the same definition of f_{t_j}.

Initialization: Set mean and covariance for each target t to values $\mathbf{m}_{0,t}, \mathbf{P}_{0,t}$ such that the prior state distribution is $\mathbf{x}_{0,t} \sim N(\mathbf{m}_{0,t}, \mathbf{P}_{0,t})$.

Prediction:
The prediction step is the same as with standard Kalman Filter for each target t separately:

$$\begin{aligned} \mathbf{m}^-_{k,t} &= \mathbf{A}_{k-1,t}\mathbf{m}_{k-1,t} \\ \mathbf{P}^-_{k,t} &= \mathbf{Q}_{k-1,t} + \mathbf{A}_{k-1,t}\mathbf{P}_{k-1,t}\mathbf{A}^T_{k-1,t} \end{aligned} \tag{6.41}$$

Gating: Select set of *validated measurements* for each target $\mathbf{Y}_{k,t} = \{\mathbf{y}^{(i)}_{k,t}\}, \quad i = 1, \dots, m_{k,t}$ for time step t_k by accepting only those measurements that lie inside the gate (similar to with (6.11), but for each target separately). Predicted mean $\hat{y}^-_k$ and covariance for the "true" measurements are given as

$$\hat{\mathbf{y}}^-_{k,t} = \mathbf{H}_{k,t}\mathbf{m}^-_{k,t} \tag{6.42}$$

$$\mathbf{S}_{k,t} = \mathbf{H}_{k,t}\mathbf{P}^-_{k,t}\mathbf{H}^T_{k,t} + \mathbf{R}_{k,t} \tag{6.43}$$

Update: The mean and covariance updates are defined as follows:

$$\mathbf{m}_{k,t} = \mathbf{m}^-_{k,t} + \mathbf{K}_k\mathbf{v}_k \tag{6.44}$$

$$\mathbf{P}_{k,t} = \beta_{k,0}\mathbf{P}^-_{k,t} + (1 - \beta_{k,0})\mathbf{P}^c_{k,t} + \tilde{\mathbf{P}}_{k,t} \tag{6.45}$$

where the Kalman Gain $\mathbf{K}_{k,t}$, combined innovation $\mathbf{v}_{k,t}$, the covariance of the correct measurement $\mathbf{P}^c_{k,t}$ and spread of the innovations term $\tilde{\mathbf{P}}_{k,t}$ are given as:

$$\mathbf{K}_{k,t} = \mathbf{P}^-_k\mathbf{H}^T_{k,t}\mathbf{S}^{-1}_{k,t} \tag{6.46}$$

$$\mathbf{v}_{k,t} = \sum_{i=1}^{m_{k,t}} \beta_{k,i,t}\mathbf{v}_{k,i,t} \tag{6.47}$$

$$\mathbf{P}^c_{k,t} = \mathbf{P}^-_{k,t} - \mathbf{K}_{k,t}\mathbf{S}_{k,t}\mathbf{K}_{k,t} \tag{6.48}$$

$$\tilde{\mathbf{P}}_{k,t} = \mathbf{K}_k\left[\sum_{i=1}^{m_{k,t}} \beta_{k,i,t}\mathbf{v}_{k,i,t}\mathbf{v}^T_{k,i,t} - \mathbf{v}_{k,i,t}\mathbf{v}^T_{k,i,t}\right]\mathbf{K}^T_{k,t} \tag{6.49}$$

Association probabilities can be calculated from (6.50).

Algorithm 16: Joint Probabilistic Data Association

- *JPDA coupled (JPDAC)* filter has the joint association event probability given as

$$p(\theta_k|\mathbf{Y}_k) \propto \frac{\phi!}{m_k} p_{\mathrm{FA}}(\phi) V^{-\phi} f_{t_{j_1},t_{j_2},\ldots}(\mathbf{y}_{k,j}, j : \tau_j = 1) \times \prod_t \left(P_{t,D}\right)^{\delta_t} \left(1 - P_{t,D}\right)^{1-\delta_t}, \tag{6.54}$$

  where $f_{t_{j_1},t_{j_2},\ldots}$ is the joint PDF of the measurements of the targets under consideration. Clutter model p_{FA} can be one of (6.26) and (6.25).

  The state consists of stacked states and the joint covariance matrix containing all the targets together with joint effects is used. See (Bar-Shalom and Li, 1995) for details.

- *JPDA with merged measurements (JPDAM)* is an extension of JPDA, which uses probabilistic model for the merged measurements from two targets. Detailed analysis and formulas are given in (Bar-Shalom and Li, 1995).

6.6 Bootstrap Filter for Multiple Target Tracking

Due to latent variable formulation used in particle filtering for cluttered case, we can easily extend it to multiple target case. Assume for simplicity that we know the number of targets, say N. Now the state vector consists of states of all the N targets:

$$\mathbf{x}_k = \begin{pmatrix} \mathbf{x}_{k,1} \\ \mathbf{x}_{k,2} \\ \vdots \\ \mathbf{x}_{k,N} \end{pmatrix}. \tag{6.55}$$

The likelihood measurement n for each target j could be, for example, given as

$$p(\mathbf{y}_{k,n}|\mathbf{x}_{k,j}, \Lambda_k) = p(\mathbf{y}_{k,n}|\mathbf{x}_{k,j}, \lambda_k) = \begin{cases} \rho_0, & \lambda_k = 0 \\ N(\mathbf{y}_{j,n}k|\mathbf{h}_i(\mathbf{x}_{k,j}), \mathbf{R}_{k,j}) & \lambda_k = j. \end{cases} \tag{6.56}$$

Each measurement likelihood depends only on the associated components in $\mathbf{x}_k$.

In case of multiple measurements, the association indicator contains separate components for each measurement. The prior probability (or likelihood in MHT) of associations may now depend on number of false alarms this indicator defines. This number of false alarms can be for example formulated as Poisson of Diffuse models as in case of PDA and JPDA.

Karlsson and Gustafsson (2001) propose a Particle Filter called SIR/MCJPDA for this multiple target tracking problem with known number of targets. They also propose using a particle filter controller, which it is not described here. The

SIR/MCJPDA method is described as the Algorithm 17. This algorithm is, in fact, the same as defined in Algorithm 15, but extended to contain multiple targets instead of only one.

Initialization: Sample N samples from prior distributions for each target j

$$\mathbf{x}_{0,j}^{(i)} \sim p(\mathbf{x}_{0,j}) \tag{6.57}$$

Prediction: Sample N new samples from

$$\mathbf{x}_k^{(i)} \sim p(\mathbf{x}_k|\mathbf{x}_{k-1}^{(i)}) \tag{6.58}$$

and corresponding latent variables from

$$\lambda_k^{(i)} \sim p(\lambda_k|\mathbf{x}_k^{(i)}) \tag{6.59}$$

Update: Evaluate importance weights

$$w_k^{(i)} \propto w_{k-1}^{(i)} p(\mathbf{y}_k|\mathbf{x}_k^{(i)}, \lambda_k^{(i)}) \tag{6.60}$$

and normalize them.
Resampling: Use the resampling scheme as in Algorithm 11.
Set $k \leftarrow k+1$ and go to step 2.

Algorithm 17: Bootstrap Filter for Multiple Targets

Hue et al. (2000) suggest that the measurement association probabilities could be also treated as random variables and propose an algorithm based on Gibbs sampling (see Section 2.4) of these probabilities.

Chapter 7

Angular Tracking

7.1 Role of Angular Tracking

In *multiple sensor tracking* systems it is often most natural to use *Cartesian* coordinates and their derivatives as the state variables. This is the most general coordinate system, which is common to all sensors and targets together. Typically, even if we did the tracking in some other coordinate system we would have to convert result to Cartesian coordinates in some point anyway. Using, for example, polar coordinates centered on one of the sensors would lead to unnecessary transformations of coordinates. This is a problem when dealing with probability distributions and their sufficient statistics representations.

Cartesian and polar coordinates themselves are equivalent in sense that we can always transform a polar coordinate point into equivalent Cartesian coordinate point and vice versa using the formulas given in Section 7.3. However, it is much harder to convert probability distributions between the coordinate systems. In tracking algorithms, probability distributions are used for representing uncertainty and this becomes an issue. We have to use either polar coordinate system or Cartesian coordinate system but we shouldn't mix them to avoid those transformations.

In tracking, linearity is good thing and non-linearity is bad, because linear systems can be solved exactly (and efficiently) using simple formulas, but non-linear systems (in general) cannot. Thus, non-linear systems are order of magnitude harder. This is why we should avoid non-linearities whenever possible. This hardness of handling non-linearities comes from the fact that linear transformations of Gaussian distributions remain Gaussian, but in non-linear transformations we lose the Gaussianity.

If we are using linear motion models and angular sensors, using *Cartesian representation* leads to following kind of model:

- Dynamic model is *linear*, because approximate Newtonian dynamics are *stochastic linear differential equations* in Cartesian coordinates.

- Measurement model is *non-linear*, because angle is related to Cartesian coordinate through $\tan^{-1}(\cdot)$.

In initialization, we will have a situation that we are actually doing *single sensor tracking* with one angular sensor. This is likely to be very hard, because posterior distribution in Cartesian coordinates is almost infinite or at least very long in one direction. The problem is that if we tried to represent it as Gaussian distribution we would have correlated Gaussian distribution having one very long non-independent direction. In this case it could be better to represent the state temporarily in polar coordinate system. This has the benefit that the long direction becomes independent direction (range) and the other component (angle) won't be disturbed by that direction.

Another advantage of using polar coordinates is that posterior distribution given measurements from only one sensor has shape of a sector in Cartesian coordinates. In polar coordinates this posterior distribution is Gaussian if we ignore the nonlinearity caused by dynamic model. Thus, in initialization stage, when dynamic model has very small effect on tracking, state distribution can be more accurately represented in polar coordinates.

If we are using linear motion models and one angular sensor, using *Polar coordinate representation* centered to the sensor leads to following kind of model:

- Dynamic model is *non-linear*, because approximate Newtonian dynamics are *stochastic non-linear differential equations* when transformed to polar coordinate system.

- Measurement model is *linear*, because angle is part of our state representation.

In initialization stage it could be more useful to be able to handle measurements exactly and use approximate model for dynamics, because the dynamics won't be anyhow useful before we have obtained enough measurements. Also, we could use simplified dynamic models in initialization stage in order to reduce number of parameters to be estimated in the beginning.

7.2 Transformations of Coordinate Systems

Assume that we are modeling dynamics of an object with generic *Ordinary Differential Equation* in form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, t) \tag{7.1}$$

where state $\mathbf{x} \in \mathbb{R}^n$ may contain, for example, position of the target together with finite number of its derivatives.

Suppose that we want represent this dynamic model in different coordinate system defined by coordinates Φ

$$\dot{\Phi} = \hat{\mathbf{f}}(\Phi, t), \tag{7.2}$$

and the transformation from coordinates Φ to $\mathbf{x}$ and its inverse are given as

$$\mathbf{x} = \mathbf{u}(\Phi) \tag{7.3}$$

$$\Phi = \mathbf{u}^{-1}(\mathbf{x}). \tag{7.4}$$

The derivation begins by differentiating the inverse transformation with respect to time:

$$\dot{\Phi} = \left(\frac{\partial \mathbf{u}^{-1}(\mathbf{x})}{\partial \mathbf{x}^T}\right) \dot{\mathbf{x}}. \tag{7.5}$$

Since $\mathbf{u}$ is invertible, we have

$$\mathbf{u}^{-1}(\mathbf{u}(\Phi)) = \Phi \tag{7.6}$$

$$\left(\frac{\partial \mathbf{u}(\Phi)}{\partial \Phi^T}\right)\left(\frac{\partial \mathbf{u}^{-1}(\mathbf{x})}{\partial \mathbf{x}^T}\right) = \mathbf{I} \tag{7.7}$$

$$\left(\frac{\partial \mathbf{u}^{-1}(\mathbf{x})}{\partial \mathbf{x}^T}\right) = \left(\frac{\partial \mathbf{u}(\Phi)}{\partial \Phi^T}\right)^{-1}, \tag{7.8}$$

where $\mathbf{x} = \mathbf{u}^{-}(\Phi)$. Using the result above and the original model, we get

$$\begin{aligned}
\dot{\Phi} &= \left(\frac{\partial \mathbf{u}^{-1}(\mathbf{x})}{\partial \mathbf{x}^T}\right) \dot{\mathbf{x}} \\
&= \left(\frac{\partial \mathbf{u}(\Phi)}{\partial \Phi^T}\right)^{-1} \mathbf{f}(\mathbf{x}, t) \\
&= \left(\frac{\partial \mathbf{u}(\Phi)}{\partial \Phi^T}\right)^{-1} \mathbf{f}(\mathbf{u}(\Phi), t).
\end{aligned} \tag{7.9}$$

So, the model can be written in form

$$\dot{\Phi} = \hat{\mathbf{f}}(\Phi, t), \tag{7.10}$$

when function $\hat{\mathbf{f}}$ is defined as

$$\hat{\mathbf{f}}(\Phi, t) = \left(\frac{\partial \mathbf{u}(\Phi)}{\partial \Phi^T}\right)^{-1} \mathbf{f}(\mathbf{u}(\Phi), t). \tag{7.11}$$

7.3 Transformation to Polar Coordinates

Suppose that we want to use polar coordinate representation of the object position. And we want represent the dynamics in terms of polar coordinates (r, θ) and their derivatives. Polar coordinate system in formed with respect to some origin (x_0, y_0), and the transformation from polar to Cartesian is

$$x = r\cos(\theta) + x_0 \tag{7.12}$$

$$y = r\sin(\theta) + y_0. \tag{7.13}$$

Note that we are using mathematical convention of polar coordinates, where zero angle is to the east from origin and the positive direction is counterclockwise around the origin.

The transformation for derivatives can be obtained by differentiating Equations (7.12) and (7.13) with respect to time

$$\dot{x} = \dot{r}\cos(\theta) - r\sin(\theta)\dot{\theta} \tag{7.14}$$

$$\dot{y} = r\cos(\theta)\dot{\theta} + \dot{r}\sin(\theta) \tag{7.15}$$

The coordinate transformation $\mathbf{x} = \mathbf{u}(\Phi)$ is given as

$$\begin{pmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{pmatrix} = \begin{pmatrix} r\cos(\theta) + x_0 \\ r\sin(\theta) + y_0 \\ \dot{r}\cos(\theta) - r\sin(\theta)\dot{\theta} \\ r\cos(\theta)\dot{\theta} + \dot{r}\sin(\theta) \end{pmatrix}. \tag{7.16}$$

The Jacobian matrix of the transformation is given as

$$\frac{\partial \mathbf{u}(\Phi)}{\partial \Phi^T} = \begin{pmatrix} \cos(\theta) & -r\sin(\theta) & 0 & 0 \\ \sin(\theta) & r\cos(\theta) & 0 & 0 \\ -\dot{\theta}\sin(\theta) & -\dot{r}\sin(\theta) & -\dot{\theta}r\cos(\theta) & \cos(\theta) - r\sin(\theta) \\ \dot{\theta}\cos(\theta) & \dot{r}\cos(\theta) & -\dot{\theta}r\sin(\theta) & \sin(\theta)r\cos(\theta) \end{pmatrix} \tag{7.17}$$

The inverse of this matrix is

$$\left(\frac{\partial \mathbf{u}(\Phi)}{\partial \Phi^T}\right)^{-1} = \begin{pmatrix} \cos(\theta) & \sin(\theta) & 0 & 0 \\ -\frac{\sin(\theta)}{r} & \frac{\cos(\theta)}{r} & 0 & 0 \\ -\dot{\theta}\sin(\theta) & \dot{\theta}\cos(\theta) & \cos(\theta) & \sin(\theta) \\ \frac{\dot{r}\sin(\theta) - \dot{\theta}r\cos(\theta)}{r^2} & -\frac{\dot{\theta}r\sin(\theta) + \dot{r}\cos(\theta)}{r^2} & -\frac{\sin(\theta)}{r} & \frac{\cos(\theta)}{r} \end{pmatrix}. \tag{7.18}$$

The transformation from Cartesian to polar coordinates is

$$r = \sqrt{(x - x_0)^2 + (y - y_0)^2} \tag{7.19}$$

$$\theta = \arctan\left(\frac{y - y_0}{x - y_0}\right). \tag{7.20}$$

By differentiating these we get

$$\dot{r} = \frac{\dot{y}(y-y_0)+\dot{x}(x-x_0)}{\sqrt{(x-x_0)^2+(y-y_0)^2}} \tag{7.21}$$

$$\dot{\theta} = \frac{\dot{y}(x-x_0)-\dot{x}(y-y_0)}{(y-y_0)^2}. \tag{7.22}$$

So, the polar initial conditions $(r(0), \theta(0), \dot{r}(0), \dot{\theta}(0))$ can be calculated from Cartesian initial conditions $(x(0), y(0), \dot{x}(0), \dot{y}(0))$ using the Equations (7.19), (7.20), (7.21) and (7.22).

7.4 Known Angular Velocity Model in Polar Coordinates

Consider 2-dimensional model for constant speed with known angular velocity

$$\ddot{x} = -a\dot{y} \tag{7.23}$$

$$\ddot{y} = a\dot{x}. \tag{7.24}$$

If state $\mathbf{x}$ is defined to contain 2 dimensional Cartesian position and velocity of the object

$$\mathbf{x} = \begin{pmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{pmatrix}, \tag{7.25}$$

the model is actually LTI model with feedback matrix

$$\mathbf{F} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -a \\ 0 & 0 & a & 0 \end{pmatrix}. \tag{7.26}$$

Now we get

$$\begin{aligned} \hat{\mathbf{f}}(\Phi, t) &= \left(\frac{\partial \mathbf{u}(\Phi)}{\partial \Phi^T}\right)^{-1} \mathbf{Fu}(\Phi) \\ &= \begin{pmatrix} \dot{r} \\ \dot{\theta} \\ \dot{\theta}^2 r - \dot{\theta} r a \\ \frac{\dot{r}a - 2\dot{r}\dot{\theta}}{r} \end{pmatrix}. \end{aligned} \tag{7.27}$$

The resulting model is given as

$$\ddot{r} = \dot{\theta}^2 r - \dot{\theta} r a \tag{7.28}$$

$$\ddot{\theta} = \frac{\dot{r}a - 2\dot{r}\dot{\theta}}{r}. \tag{7.29}$$

7.5 Nearly Constant Velocity Model in Polar Coordinates

The model in previous section was deterministic differential equation, but dynamic models are more frequently expressed as *stochastic differential equations* in form

$$\dot{\mathbf{x}} = \mathbf{F}\mathbf{x} + \mathbf{L}\mathbf{w}(t), \tag{7.30}$$

where $\mathbf{w}$ is a Gaussian white noise process:

$$\mathrm{E}[\mathbf{w}(t)] = \mathbf{0} \tag{7.31}$$
$$\mathrm{E}[\mathbf{w}(t)\mathbf{w}(t+\tau)^T] = \mathbf{Q}_c\delta(\tau). \tag{7.32}$$

In polar coordinates, this model can be expressed as

$$\dot{\Phi} = \hat{\mathbf{f}}(\Phi, t), \tag{7.33}$$

where

$$\hat{\mathbf{f}}(\Phi, t) = \left(\frac{\partial \mathbf{u}(\Phi)}{\partial \Phi^T}\right)^{-1} \mathbf{F}\mathbf{u}(\Phi) + \left(\frac{\partial \mathbf{u}(\Phi)}{\partial \Phi^T}\right)^{-1} \mathbf{L}\mathbf{w}(t). \tag{7.34}$$

As on example, consider *nearly constant velocity model*, which can be obtained from Equations (7.23) and (7.24) by setting angular velocity to zero $a = 0$, and using a Wiener noise model for velocities:

$$\begin{aligned} \ddot{x} &= w_1(t), \qquad w_1(t) \sim N(0, q) \\ \ddot{y} &= w_2(y), \qquad w_2(t) \sim N(0, q), \end{aligned} \tag{7.35}$$

which is model in form of Equation (7.30) with

$$\mathbf{x} = \begin{pmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{pmatrix} \qquad \mathbf{F} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{7.36}$$

and

$$\mathbf{L} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad \mathbf{Q}_c = \begin{pmatrix} q & 0 \\ 0 & q \end{pmatrix}. \tag{7.37}$$

In polar coordinates, the nearly constant velocity model in Equation (7.35) is given as

$$\begin{aligned} \ddot{r} &= \dot{\theta}^2 r + \sin(\theta) w_2(t) + \cos(\theta) w_1(t) \\ \ddot{\theta} &= -\frac{2\dot{r}\dot{\theta}}{r} + \frac{\cos(\theta)}{r} w_2(t) - \frac{\sin(\theta)}{r} w_1(t), \end{aligned} \tag{7.38}$$

with

$$\begin{aligned} \mathrm{E}[\mathbf{w}(t)] &= \mathbf{0} \\ \mathrm{E}[\mathbf{w}(t)\mathbf{w}(t+\tau)^T] &= \mathbf{Q}_c\delta(\tau). \end{aligned} \tag{7.39}$$

Chapter 8

Negative Information Modeling

8.1 Overview of Negative Information Modeling

In a target tracking system physical sensors report measurements only when they receive some kind of signal, which can be further processed into a measurement. In the single-sensor case all we can do is to use these signal-induced measurements. However, when there are multiple sensors measuring from the same origin (e.g., the radar of a target), and some sensors can detect this signal and some cannot, our information is increased by knowing the fact that some sensors could not detect the target. (We need initially multiple sensors since if no sensor detects the target, we do not know that there is a target.) This is called negative information, and from it we know that target must be in such a place or orientation that this sensor cannot see it. The inability to observe the target can, for example, be caused by the target being outside the sensor's surveillance area in the state space or its radar pointing into the wrong direction. More generally, the target's position in the state space (position, direction, derivatives, etc.) is such that the sensor cannot see it.

The physical sensors observing the target do not generate the negative information, since they cannot guess what they should see. They don't know anything about the targets, nor which target their measurement originated from. But after the normal or positive information measurement processing and data association has been performed, it is possible to detect which sensors could have reported measurements, and did not. If such sensors exist, we can generate artificial negative information measurements which indicate that the target is likely to be outside the surveillance areas of these sensors.

The negative information model consist of two parts:

1. **Detecting** the negative information and generating the negative information measurements.

2. **Updating** the system state estimate using the information in negative information measurements.

8.2 Detecting the Negative Information

Here we shall present a conceptual example of how we could, in principle, detect negative information. However, the reader should understand that the whole problem is very complex. In order to detect negative information, we would have to be able to fully model the detection generation procedure connected to specific sensors and targets. There simply is no generic way of doing this. However, sensor experts (sensor engineers) may have such practical knowledge that they could help in creating these detection generation and negative information detection models.

On a general level there are two kinds of sensor-measurement pairs. The first is perhaps the more typical one, in which the measurement is generated by a sensor and the measurement is dependent only on the location and speed of the target. Examples of this are radar sensors, human observers and such. The target cannot prevent the measurement, and thus we gain a lot from information from the absence of measurements - if a sensor cannot see a target that is known to exist, we know that the target is probably occluded to the sensor.

In the second type the measurement is obtained by interpreting something that the target does. For example, underwater or radio listening stations can pick up sonar and radio pulses. With this kind of measurements it is far more difficult to obtain useful negative information. If a sensor does not receive measurements, it might simply be because the target is silent, or then the target might be occluded to the sensor, as before. Thus, in this case we gain less information from the absence of measurements - only if we somehow know that the target is transmitting and should be detected, can we deduce that it is occluded to a sensor which does not report measurements.

The following scenario illustrates how negative information could generally be detected. Consider Figure 8.1, where there are two sensors getting measurements from a moving target. The sensors have finite surveillance areas. Initially, the target is inside the surveillance areas of both sensors and we get measurements from both of them. After a while the target leaves the surveillance area of Sensor 2 and only Sensor 1 continues to report measurements.

The signal detection streams as function of time are shown in Figure 8.2. From the figure we can see that we get detections from both the sensors on quite regular intervals. These detections don't come from both the sensors at the same time, but there is clearly a stream of detections coming from both sensors. Suddenly, approximately at time $t = 6$, Sensor 2 stops reporting measurements. Looking at Figure 8.1 we can see the reason - this is the time when the target exits the surveillance area of Sensor 2.

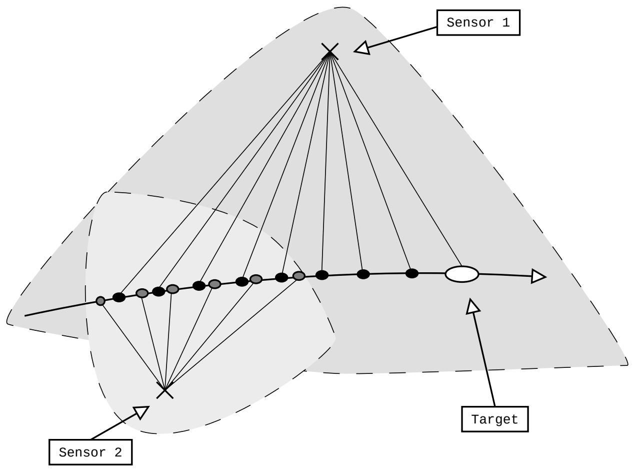


Figure 8.1: *In the beginning the target is inside the surveillance area of both sensors 1 and 2. When it leaves the surveillance area of sensor 2, only Sensor 1 continues to report measurements.*

It is possible to detect the time instance when Sensor 2 loses the target from the detections in Figure 8.2 by building a suitable stochastic model for the sensor detection time series. Because we have detected that there clearly is a target (because Sensor 1 sees it) and still Sensor 2 cannot see it, we can generate artificial negative information measurements for Sensor 2. These measurements are target-sensor pairs, which indicate that this sensor cannot see this target. Another possible reason would be that the signal has really ended (e.g., the target has been destroyed), but this reason has been eliminated by the fact the other sensor still sees the target.

This same idea can be easily generalized to multiple sensors. If at least one sensor receives a signal, we know that there is a signal. If some other sensor does not receive the signal, we can possibly use this as negative information. Of course, if the sensor is of the kind where the target's actions influence the measurement, one has to be careful - there might be several reasons for this loss of signal. For example, the sensor can be turned off or there might be an unexpected (unmodeled) phenomenon which blocks the signal inside the sensor's surveillance area. It might even be possible to disturb the negative information estimation algorithm by selectively blocking the signal from some sensors, which could lead to wrong position estimates of targets. However, by carefully designing the system this can probably be avoided.

The generalization to multiple targets is also possible. In order to generate the negative information measurements we must know the data associations of

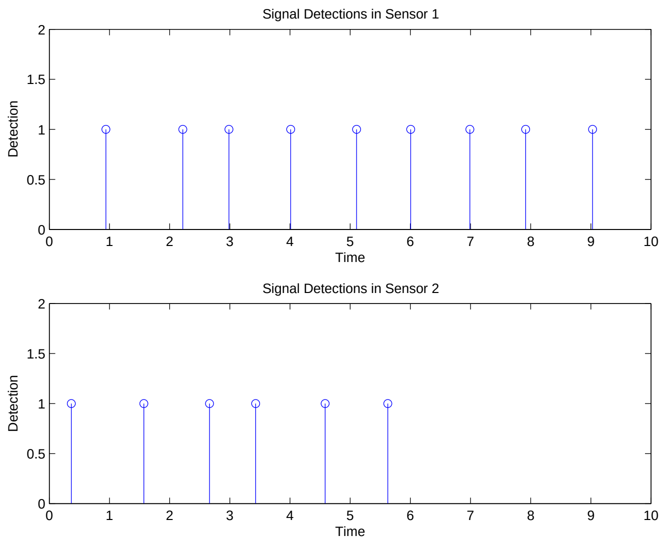


Figure 8.2: *Signal detections in sensors 1 and 2, when the target is moving as in Figure 8.1. When the target exists the surveillance area of Sensor 2, the regular detection stream suddenly ends.*

"positive" measurements. After that we can check which of the sensors have produced measurements from specific targets and which haven't. This association knowledge requirement means that negative information measurements depend on association hypotheses - we have different negative information measurements for different associations. If we use the Monte Carlo method for data association, we should estimate the sensor-target time series for each association hypothesis separately.

8.3 Using the Negative Information Measurements

In practice negative information can be utilized by generating negative information measurements. A negative information measurement means that at a given point of time, we know that a sensor should have reported a measurement, and did not. That is, negative information is triggered, as described in Section 8.2. As will be shown in the following, the possibility of negative information also affects the likelihood of positive measurements (Stone et al., 1999).

Let $S \subset \Re^n$ be the state space of the target. We define our measurement space as $\hat{M} = M \cup \{\phi\}$, where $M \subset \Re^m$ is the space of positive measurements and ϕ denotes the event "no measurement". The sensor model defines a probability $\hat{p}(y|x)$ for all target states $x \in S$, where $y \in \hat{M}$. Thus the distribution over the whole M is

$$p(y|x) = \begin{cases} (1 - p(\phi|x))\hat{p}(y|x), & y \neq \phi \\ p(\phi|x), & y = \phi. \end{cases} \tag{8.1}$$

Instead of the probability of no detection $p(\phi|x)$ we can define the probability of detection, $p_d(x) = 1 - p(\phi|x)$, in which case Equation 8.1 becomes

$$p(y|x) = \begin{cases} p_d(x)\hat{p}(y|x), & y \neq \phi \\ 1 - p_d(x), & y = \phi. \end{cases} \tag{8.2}$$

One can see that negative information affects also the likelihood function of positive measurements - if a measurement is received, we know that the target is not located outside the area visible to the sensor. (However, it should be noted that this effect is relatively minor compared to how negative information affects the posterior when the target is not seen.) For multiple sensors with independent detections the joint likelihood function is

$$L(x) = \prod_{i=1}^{N_d} p_d^{(i)}(x)\hat{p}^{(i)}(y^{(i)}|x) \prod_{j=1}^{N_n} (1 - p_d^{(j)}(x)), \tag{8.3}$$

where $y^{(i)}$ is the ith measurement, $p^{(i)}$ the measurement model of the ith sensor, N_d the number of sensors that detect the target and N_n the number of sensor that miss the detection.

The negative information likelihood function $p(\phi|x)$ can be defined in many ways. The simplest possibility is to have the negative information likelihood be uniform in the occluded area and zero in the observed area whenever the sensor does not receive measurements. That is, the state distributions are truncated according to the sensor's surveillance area. The negative information likelihood can thus be written as

$$p(\phi|x) = \begin{cases} 0 & \text{if, } x \in A \\ 1 & \text{if, } x \notin A, \end{cases} \tag{8.4}$$

where A is the part of the state space observed by the sensor. A simple 1D negative information scenario illustrates this kind of likelihood in Figures 8.3, 8.4, and 8.5. The tracking is done by a regular bootstrap filter (Algorithm 11) using a simple $\ddot{x} = 0$ dynamic model.

The likelihood function presented above assumes that the probability of detection is 1. A more realistic likelihood is not zero in the observed area, but rather nonzero at some uniform level, reflecting the possibility of the sensor missing the target. Naturally, this level should be lower in the areas visible to the sensor than the occluded areas. For example, if the probability of no detection is 0.1 and no measurements are reported, with uniform distributions the (unnormalized) likelihood should be 0.1 inside the visible area and 0.9 outside the visible area. If a measurement is received, the situation is reversed - the positive likelihoods of the

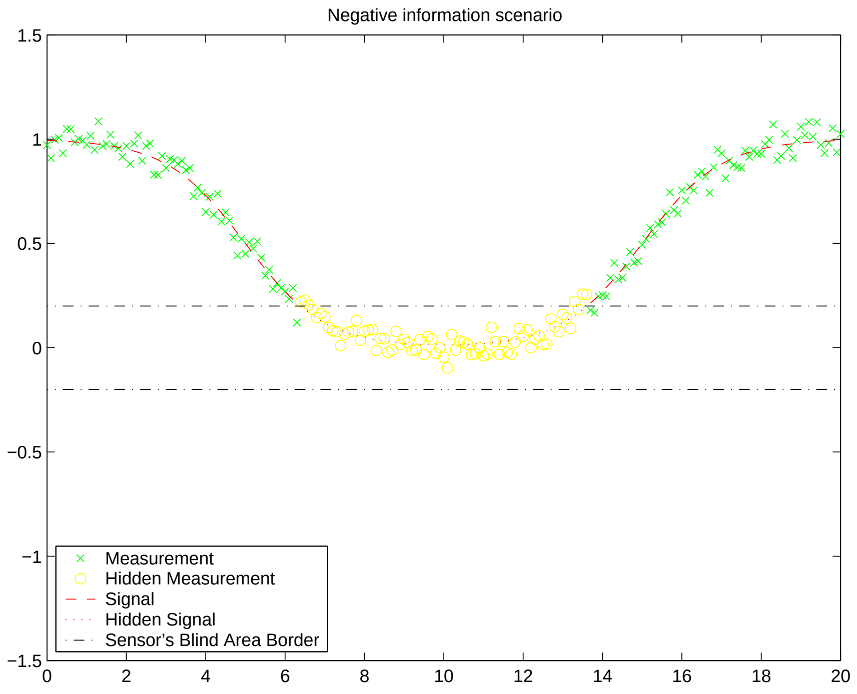


Figure 8.3: *Negative information modeling scenario. The light measurements in the middle (between the dashed lines) are assumed to be missing, because sensor is unable to measure the phenomenon when signal is in area* $[-0.2, 0.2]$.

particles inside the observed area are multiplied by 0.9, while those outside the area receive the weight 0.1. This kind of likelihood is illustrated in Figure 8.6. The figure shows the likelihood field when no sensor receives measurements.

In reality the observed areas are not clearly defined circles with uniform probabilities of detection. Common sense dictates that most sensors detect targets better when they are close to the sensor. Thus a better approximation of reality could, for example, be a radial Gaussian detection function, that is, target detection would fail with an increasing probability as distance from the sensor grows. However, at this stage of development the uniform distributions are probably accurate enough.

We can directly implement a particle filter once the negative information likelihood function has been decided upon. The basic filter algorithm is unchanged, with prediction step is carried out as usual. After this the weights are updated according to the modified negative information likelihood function and resampling is performed. Implementing a Kalman filter with negative information is a little more complicated, as the likelihood distribution is not Gaussian or even a mixture of Gaussians - also the positive information likelihood is affected, as explained before. As the shapes of the areas occluded to sensors are usually quite compli-

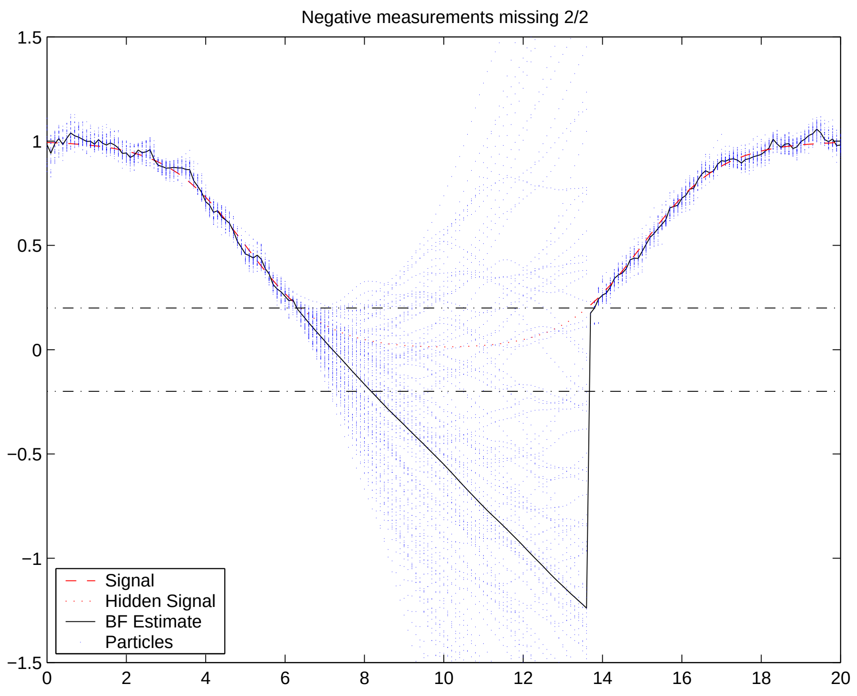


Figure 8.4: *Tracking result of particle filter, when the lost measurements are modeled as missing. That is, if no measurement arrive, we simply predict the new state and perform no measurement update. The distribution of the location quickly widens as only the prediction model can be used. Note how the mean of the distribution moves in a linear fashion.*

cated, in most cases it is probably impossible to perform the measurement update in closed form even when the negative information likelihood is of the simple, uniform type. One way to solve this problem could be to transform the Kalman filter into a particle filter for the measurement update, that is, sample a set from the Kalman predictive distribution, update the sample weights as before with the particle filter, and return to the Kalman filter by computing the posterior weighted mean and covariance of the samples.

The main advantage of using negative information is that the distribution of states is physically feasible and more realistic when the target is not visible. An example of this is the tracking of a single person : if we see a person go into a room and not come out, after five minutes we can be sure that the person is still in the room if all exits have been observed. However, the precise location of the person inside the room is very vague - in practice the distribution of the person's location is very close to uniform. If the occluded areas are small in volume and regular in shape, then negative information can be used to aid the tracking process, if the

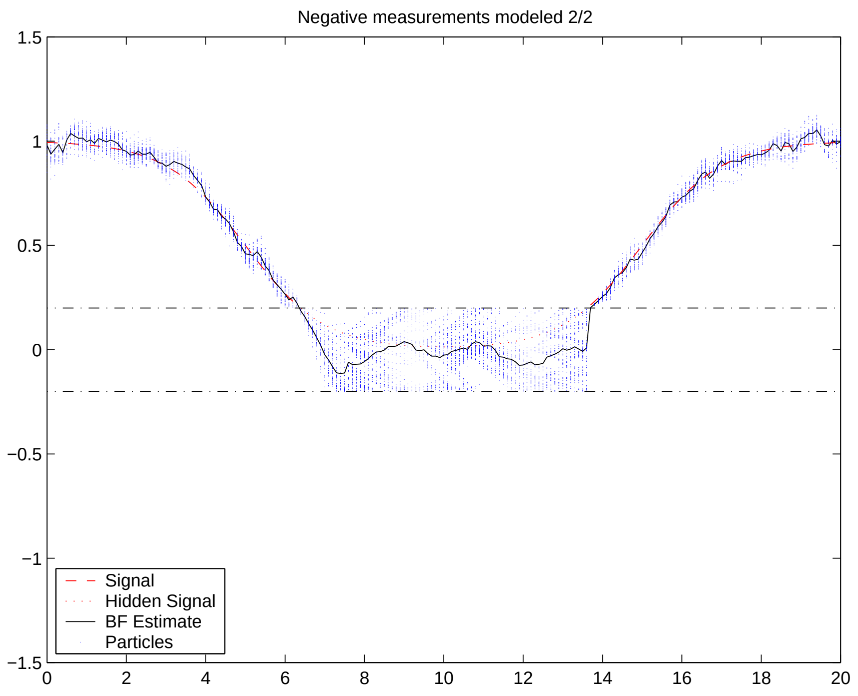


Figure 8.5: *Tracking result of particle filter, when the lost measurements are modeled as being in area* $[-0.2, 0.2]$. *Now the distribution stays inside the boundaries and does not explode.*

tracked person enters a narrow corridor at a constant speed, it is highly probable that the person will exit the corridor at the other end.

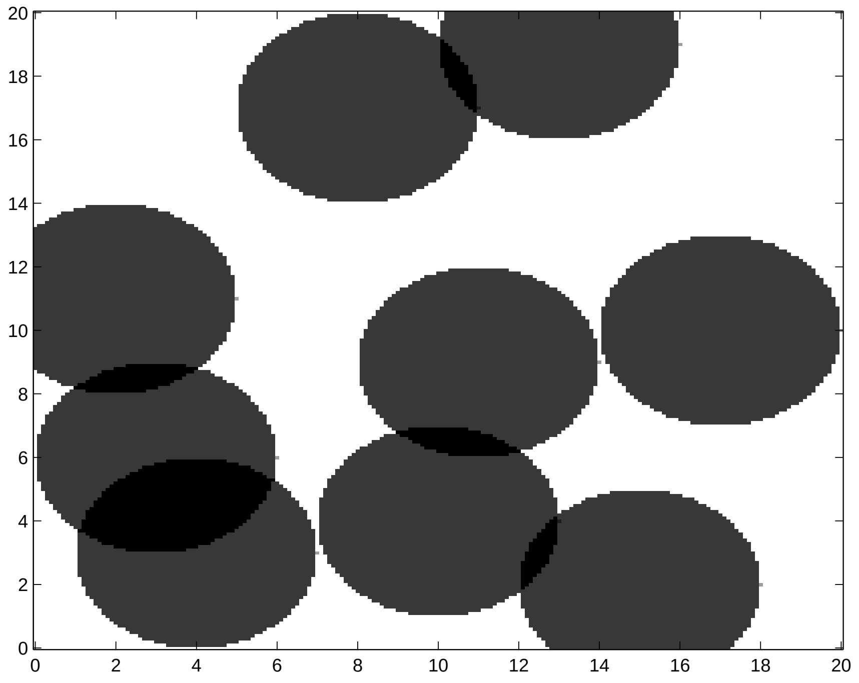


Figure 8.6: *Sample 2D negative information likelihood field, when no sensor receives measurements and the probability of a sensor missing a target is 0.1. Each sensor observes a circular area around it, i.e. in the image the sensors are located in the centers of the dark circles. Note how the lowest likelihood occurs at areas which two sensors can see.*

Chapter 9

Summary

In this document we have reviewed the most commonly used probabilistic methods in multiple target tracking. The document could be summarized as follows:

- **Monte Carlo methods** are useful tools for estimation and representation of posterior distributions in Bayesian inference. Because the probabilistic filtering problem belongs to class of Bayesian models, Monte Carlo methods can be applied to the filtering problem also. The sequential nature of the model complicates the problem, but sequential Monte Carlo or particle filtering methods have been developed for efficient estimation in dynamic models.

- **Kalman filtering methods** such as linear Kalman filter, extended Kalman filter and unscented Kalman filter are (or can be interpreted as) Bayesian estimators that sequentially form Gaussian approximations of the state posterior distribution. The methods can be directly used as sub-estimators in multiple target tracking algorithms.

- **Dynamics of targets** are most naturally modeled as linear or non-linear stochastic differential equations. This is the most general way of solving the problem of asynchronous sensors that produce measurements on irregular intervals.

- **Multiple target tracking** can be formulated as probabilistic filtering model, where data association ambiguity is modeled using a latent variable, which labels the measurement origin. The problem of multiple target tracking reduces to problem of developing methods for representing the joint posterior distribution of states and latent variables as accurately as possible.

- **PDA and JPDA** are multiple target tracking algorithms that represent the state posterior as set of Gaussian distributions, such that each Gaussian distribution represents one target.

- **MHT** is a multiple target tracking algorithm that forms hypotheses of different data associations and calculates relative probabilities of them. Based on these calculated probabilities it forms a set of most probable data association histories, which is propagated in time and updated using the sensor measurements.

- **Sequential Monte Carlo methods** can be directly applied to estimation of the probabilistic filtering model that represents the multiple target tracking problem. These methods can be also used for handling the negative information, where PDA, JPDA and MHT cannot be directly used. However, the pure bootstrap filter is unlikely to be efficient as such, and good importance distributions and possibility of Rao-Blackwellization should be evaluated.

Appendix A

Derivations

A.1 Derivation of Kalman Filter

Assume a model

$$\mathbf{x}_k = \mathbf{A}_{k-1}\mathbf{x}_{k-1} + \mathbf{B}_{k-1}\mathbf{u}_{k-1} + \mathbf{q}_{k-1} \tag{A.1}$$
$$\mathbf{y}_k = \mathbf{H}_k\mathbf{x}_k + \mathbf{r}_k, \tag{A.2}$$

where

$$\mathbf{q}_{k-1} \sim N(\mathbf{0}, \mathbf{Q}_{k-1}) \tag{A.3}$$
$$\mathbf{r}_k \sim N(\mathbf{0}, \mathbf{R}_k). \tag{A.4}$$

Matrices $\mathbf{A}_{k-1}$, $\mathbf{B}_{k-1}$, $\mathbf{H}_k$, $\mathbf{Q}_{k-1}$, $\mathbf{R}_k$ and input vector $\mathbf{u}_{k-1}$ are assumed known for all $k > 0$. The initial configuration defined by parameters $\mathbf{m}_0$ and $\mathbf{P}_0$ is also assumed to be known.

Equations (A.1) – (A.4) can be written in alternative notation as

$$p(\mathbf{x}_k \mid \mathbf{x}_{k-1}) = N(\mathbf{x}_k \mid \mathbf{A}_{k-1}\mathbf{x}_{k-1} + \mathbf{B}_{k-1}\mathbf{u}_{k-1}, \mathbf{Q}_{k-1}) \tag{A.5}$$
$$p(\mathbf{y}_k \mid \mathbf{x}_k) = N(\mathbf{y}_k \mid \mathbf{H}_k\mathbf{x}_k, \mathbf{R}_k). \tag{A.6}$$

The prior distribution is then

$$p(\mathbf{x}_0) = N(\mathbf{x}_0 \mid \mathbf{m}_0, \mathbf{P}_0). \tag{A.7}$$

Kalman Filter equations can be derived by induction. First assume that prior distribution is exactly Gaussian as given in Equation (A.7). Then assume that after $k-1$ steps we have Gaussian state distribution given as

$$p(\mathbf{x}_{k-1} \mid \mathbf{y}_{1:k-1}) = N(\mathbf{x}_{k-1} \mid \mathbf{m}_{k-1}, \mathbf{P}_{k-1}). \tag{A.8}$$

The predictive state distribution (distribution before just before measurement) for step k is

$$\begin{aligned} p(\mathbf{x}_k \mid \mathbf{y}_{1:k-1}) &= \int p(\mathbf{x}_k \mid \mathbf{x}_{k-1})\, p(\mathbf{x}_{k-1} \mid \mathbf{y}_{1:k-1})\, \mathrm{d}\mathbf{x}_{k-1} \\ &= \int N(\mathbf{x}_k \mid \mathbf{A}_{k-1}\mathbf{x}_{k-1} + \mathbf{B}_{k-1}\mathbf{u}_{k-1}, \mathbf{Q}_{k-1}) \\ &\quad \times N(\mathbf{x}_{k-1} \mid \mathbf{m}_{k-1}, \mathbf{P}_{k-1})\, \mathrm{d}\mathbf{x}_{k-1}. \end{aligned} \tag{A.9}$$

The integral (A.9) can be calculated in closed form and since this integral simply integrates over one parameter or Gaussian distribution, the result is Gaussian also.

Quite intuitive way to derive the parameters of this Gaussian is the following:

1. Assume that we have state distribution

$$\mathbf{x}_{k-1} \sim N(\mathbf{m}_{k-1}, \mathbf{P}_{k-1}). \tag{A.10}$$

2. The distribution of $\mathbf{x}_k^* = \mathbf{A}_{k-1}\mathbf{x}_{k-1}$ follows from elementary rules for expected values and covariances

$$E[\mathbf{x}_k^*] = E[\mathbf{A}_{k-1}\mathbf{x}_{k-1}] = \mathbf{A}_{k-1}E[\mathbf{x}_{k-1}] \tag{A.11}$$

$$Cov[\mathbf{x}_k^*] = Cov[\mathbf{A}_{k-1}\mathbf{x}_{k-1}] = \mathbf{A}_{k-1}Cov[\mathbf{x}_{k-1}]\mathbf{A}_{k-1}^T. \tag{A.12}$$

3. We still have the deterministic input and process noise term. Since the input is deterministic, it doesn't affect covariance at all, only the mean. Process noise in turn has mean zero and thus doesn't affect mean at all. Thus, we have:

$$E[\mathbf{x}_k] = E[\mathbf{x}_k^*] + \mathbf{B}_{k-1}\mathbf{u}_{k-1} = \mathbf{A}_{k-1}E[\mathbf{x}_{k-1}] + \mathbf{B}_{k-1}\mathbf{u}_{k-1} \tag{A.13}$$

$$Cov[\mathbf{x}_k] = Cov[\mathbf{x}_k^*] + \mathbf{Q}_{k_1} = \mathbf{A}_{k-1}Cov[\mathbf{x}_{k-1}]\mathbf{A}_{k-1}^T + \mathbf{Q}_{k-1}. \tag{A.14}$$

Now, using the notation $\mathbf{m}_{k-1} = E[\mathbf{x}_{k-1}]$ and $\mathbf{P}_{k-1} = Cov[\mathbf{x}_{k-1}]$ we have

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:k-1}) = N(\mathbf{x}_k \mid \mathbf{m}_k^-, \mathbf{P}_k^-), \tag{A.15}$$

where

$$\mathbf{m}_k^- = \mathbf{A}_{k-1}\mathbf{m}_{k-1} + \mathbf{B}_{k-1}\mathbf{u}_{k-1} \tag{A.16}$$

$$\mathbf{P}_k^- = \mathbf{A}_{k-1}\mathbf{P}_{k-1}\mathbf{A}_{k-1}^T + \mathbf{Q}_{k-1}. \tag{A.17}$$

Equations (A.16) and (A.17) are in fact the first two Kalman Filter equations. They are also known as the prediction step of Kalman Filter for obvious reasons.

This minus - is there to indicate that we didn't take measurement $\mathbf{y}_k$ into account yet.

The posterior distribution given the predictive distribution above can be calculated as

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:k}) \propto p(\mathbf{y}_k \mid \mathbf{x}_k\ )\ p(\mathbf{x}_k \mid \mathbf{y}_{1:k-1}) \tag{A.18}$$

$$\propto N(\mathbf{y}_k \mid \mathbf{H}_k\mathbf{x}_k, \mathbf{R}_k) N(\mathbf{x}_k \mid \mathbf{m}_k^-, \mathbf{P}_k^-). \tag{A.19}$$

We know that distribution $p(\mathbf{x}_k \mid \mathbf{y}_{1:k})$ must be Gaussian

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:k}) = N(\mathbf{x}_k \mid \mathbf{m}_k, \mathbf{P}_k), \tag{A.20}$$

since the product of two Gaussian distribution is always Gaussian. Since the mean of Gaussian distribution is the same as maximum of the distribution, the mean is actually

$$E[\mathbf{x}_k] = \arg\max_{\mathbf{x}_k} N(\mathbf{y}_k \mid \mathbf{H}_k\mathbf{x}_k, \mathbf{R}_k) N(\mathbf{x}_k \mid \mathbf{m}_k^-, \mathbf{P}_k^-). \tag{A.21}$$

This maximum is the same as minimum of negative logarithm of this function

$$E(\mathbf{x}_k) = \frac{1}{2}(\mathbf{y}_k - \mathbf{H}_k\mathbf{x}_k)^T \mathbf{R}_k^{-1}(\mathbf{y}_k - \mathbf{H}_k\mathbf{x}_k) + \frac{1}{2}(\mathbf{x}_k - \mathbf{m}_k^-)^T (\mathbf{P}^-{}_k)^{-1}(\mathbf{x}_k - \mathbf{m}_k^-). \tag{A.22}$$

The derivative of this energy function is

$$\frac{\partial E(\mathbf{x}_k)}{\partial \mathbf{x}_k} = \mathbf{H}_k^T \mathbf{R}_k^{-1}(\mathbf{H}_k\mathbf{x}_k - \mathbf{y}_k) + (\mathbf{P}_k^-)^{-1}(\mathbf{x}_k - \mathbf{m}_k^-). \tag{A.23}$$

The maximum is where this gradient is zero:

$$\mathbf{H}_k^T \mathbf{R}_k^{-1}(\mathbf{H}_k\mathbf{x}_k - \mathbf{y}_k) + (\mathbf{P}_k^-)^{-1}(\mathbf{x}_k - \mathbf{m}_k^-) = 0. \tag{A.24}$$

The solution is

$$\mathbf{x}_k = (\mathbf{H}_k^T \mathbf{R}_k^{-1}\mathbf{H}_k + (\mathbf{P}_k^-)^{-1})^{-1}(\mathbf{H}_k^T \mathbf{R}_k^{-1}\mathbf{y}_k + (\mathbf{P}_k^-)^{-1}\mathbf{m}_k^-). \tag{A.25}$$

We may now use the matrix inversion lemma

$$(\mathbf{H}_k^T \mathbf{R}_k^{-1}\mathbf{H}_k + (\mathbf{P}_k^-)^{-1})^{-1} = (\mathbf{P}_k^- - \mathbf{P}_k^-\mathbf{H}_k^T(\mathbf{R}_k + \mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T)^{-1}\mathbf{H}_k\mathbf{P}_k^-), \tag{A.26}$$

and

$$\mathbf{x}_k = (\mathbf{P}_k^- - \mathbf{P}_k^-\mathbf{H}_k^T(\mathbf{R}_k + \mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T)^{-1}\mathbf{H}_k\mathbf{P}_k^-)(\mathbf{H}_k^T \mathbf{R}_k^{-1}\mathbf{y}_k + (\mathbf{P}_k^-)^{-1}\mathbf{m}_k^-) \tag{A.27}$$

$$= \mathbf{P}_k^-\mathbf{H}_k^T \left[\mathbf{I} - (\mathbf{R}_k + \mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T)^{-1}\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T\right]\mathbf{R}_k^{-1}\mathbf{y}_k \tag{A.28}$$

$$+ \mathbf{m}_k^- - \mathbf{P}_k^-\mathbf{H}_k^T(\mathbf{R}_k + \mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T)^{-1}\mathbf{H}_k\mathbf{m}_k^-. \tag{A.29}$$

The next trick is to use the identity

$$\mathbf{I} = (\mathbf{R}_k + \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T)^{-1} (\mathbf{R}_k + \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T) \tag{A.30}$$

to simplify the coefficient of $\mathbf{y}_k$ to get

$$\begin{aligned}
\mathbf{x}_k &= \mathbf{P}_k^- \mathbf{H}_k^T [(\mathbf{R}_k + \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T)^{-1} (\mathbf{R}_k + \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T) && \text{(A.31)}\\
&\quad - (\mathbf{R}_k + \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T)^{-1} \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T] \mathbf{R}_k^{-1} \mathbf{y}_k && \text{(A.32)}\\
&\quad + \mathbf{m}_k^- - \mathbf{P}_k^- \mathbf{H}_k^T (\mathbf{R}_k + \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T)^{-1} \mathbf{H}_k \mathbf{m}_k^- && \text{(A.33)}\\
&= \mathbf{P}_k^- \mathbf{H}_k^T \left[(\mathbf{R}_k + \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T)^{-1} \mathbf{R}_k \right] \mathbf{R}_k^{-1} \mathbf{y}_k && \text{(A.34)}\\
&\quad + \mathbf{m}_k^- - \mathbf{P}_k^- \mathbf{H}_k^T (\mathbf{R}_k + \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T)^{-1} \mathbf{H}_k \mathbf{m}_k^- && \text{(A.35)}\\
&= \mathbf{m}_k^- + \mathbf{P}_k^- \mathbf{H}_k^T (\mathbf{R}_k + \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T)^{-1} \left[\mathbf{y}_k - \mathbf{H}_k \mathbf{m}_k^- \right]. && \text{(A.36)}
\end{aligned}$$

We may conclude that mean of distribution in Equation (A.20) is

$$\mathbf{m}_k = \mathbf{m}_k^- + \mathbf{K}_k \left[\mathbf{y}_k - \mathbf{H}_k \mathbf{m}_k^- \right], \tag{A.37}$$

where

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}_k^T (\mathbf{R}_k + \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T)^{-1}. \tag{A.38}$$

The covariance of distribution in Equation (A.20) can be calculated from the inverse Hessian of energy function in Equation (A.22). The Hessian is

$$\frac{\partial^2 E(\mathbf{x}_k)}{\partial \mathbf{x}_k^2} = \mathbf{H}_k^T \mathbf{R}_k^{-1} \mathbf{H}_k + (\mathbf{P}_k^-)^{-1}, \tag{A.39}$$

Using the matrix inversion lemma we get

$$\begin{aligned}
\left(\frac{\partial^2 E(\mathbf{x}_k)}{\partial \mathbf{x}_k^2} \right)^{-1} &= \left(\mathbf{H}_k^T \mathbf{R}_k^{-1} \mathbf{H}_k + (\mathbf{P}_k^-)^{-1} \right)^{-1} && \text{(A.40)}\\
&= \mathbf{P}_k^- - \mathbf{P}_k^- \mathbf{H}_k^T (\mathbf{R}_k + \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T)^{-1} \mathbf{H}_k \mathbf{P}_k^- && \text{(A.41)}\\
&= \mathbf{P}_k^- - \mathbf{K}_k \mathbf{H}_k \mathbf{P}_k^-. && \text{(A.42)}
\end{aligned}$$

If we define innovation mean and covariance as

$$\begin{aligned}
\mathbf{v}_k &= \mathbf{y}_k - \mathbf{H}_k \mathbf{m}_k^- && \text{(A.43)}\\
\mathbf{S}_k &= \mathbf{R}_k + \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T, && \text{(A.44)}
\end{aligned}$$

we get alternative form

$$\left(\frac{\partial^2 E(\mathbf{x}_k)}{\partial \mathbf{x}_k^2} \right)^{-1} = \mathbf{P}_k^- - \mathbf{K}_k \mathbf{S}_k \mathbf{K}_k^T, \tag{A.45}$$

which is numerically more stable, because of the explicit symmetry. Thus, the resulting equations for posterior distribution parameters are

$$\begin{aligned}
\mathbf{v}_k &= \mathbf{y}_k - \mathbf{H}_k \mathbf{m}_k^- && \text{(A.46)}\\
\mathbf{S}_k &= \mathbf{R}_k + \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T && \text{(A.47)}\\
\mathbf{K}_k &= \mathbf{P}_k^- \mathbf{H}_k^T \mathbf{S}_k^{-1} && \text{(A.48)}\\
\mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k \mathbf{v}_k && \text{(A.49)}\\
\mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k \mathbf{S}_k \mathbf{K}_k^T. && \text{(A.50)}
\end{aligned}$$

A.2 Propagation Equations for Linear Langevin Equations

Linear stochastic differential equation is a special case of Langevin Equation (4.1):

$$\dot{\mathbf{x}} = \mathbf{F}(t)\mathbf{x} + \mathbf{L}(t)\mathbf{w}(t), \tag{A.51}$$

where $\mathbf{w}(t)$ is a Gaussian white noise process with zero mean and spectral density $\mathbf{Q}_c(t)$. This model can be written using Einstein's summation convention as

$$\dot{x}_i = F_{ij}(t)x_j + L_{ik}(t)w_k(t). \tag{A.52}$$

The Gaussian noise process $w(t)_i$ has moments

$$\begin{aligned}
E[w_i(t)] &= 0\\
E[w_i(t)w_j(t')] &= q_{ij}(t)\delta(t-t'),
\end{aligned} \tag{A.53}$$

where $q_{ij} = q_{ji}$. We would like to find the solution to this equation using the Fokker-Planck-Kolmogorov equations. We first determine the diffusion coefficients

$$\begin{aligned}
D_i^{(1)}(\mathbf{x},t) &= F_{ij}(t)x_j && \text{(A.54)}\\
D_{ij}^{(2)}(\mathbf{x},t) &= L_{ik}(t)Q_{c,kk}(t)L_{jk}(t), && \text{(A.55)}
\end{aligned}$$

which can be written in vector form as

$$\begin{aligned}
\mathbf{D}^{(1)}(\mathbf{x},t) &= \mathbf{F}(t)\mathbf{x} && \text{(A.56)}\\
\mathbf{D}^{(2)}(\mathbf{x},t) &= \mathbf{L}(t)\mathbf{Q}_c(t)\mathbf{L}(t)^T. && \text{(A.57)}
\end{aligned}$$

The corresponding Fokker-Planck-Kolmogorov equation has now form

$$\frac{\partial p}{\partial t} = -\frac{\partial}{\partial x_i}\left(F_{ij}x_j p\right) + \frac{1}{2}L_{ik}Q_{c,kk}L_{jk}\frac{\partial^2 p}{\partial x_i \partial x_j}, \tag{A.58}$$

where we have dropped the time dependency from F's and L's for notational convenience. Time dependency is still there.

Taking Fourier transformations of both sides with respect to $\mathbf{x}$ we get (all $\partial/\partial x_i$'s are replaced with $\mathrm{i}\omega_i$ and x_i's with $\mathrm{i}\partial/\partial\omega_i$):

$$\frac{\partial \tilde{p}}{\partial t} = \omega_i F_{ij} \frac{\partial \tilde{p}}{\partial \omega_j} - \frac{1}{2} L_{ik} Q_{c,kk} L_{jk} \omega_i \omega_j \tilde{p}. \tag{A.59}$$

Fourier transformation $\tilde{p}(\omega)$ of (one dimensional) probability distribution is also called the characteristic function of probability distribution $p(x)$. This is because the moments of distribution are given by derivatives of Fourier transformation as

$$E[x^n] = \frac{1}{\mathrm{i}^n} \left. \frac{\mathrm{d}\tilde{p}(\omega)}{\mathrm{d}\omega} \right|_{\omega=0}. \tag{A.60}$$

For multidimensional distributions the moments are much more complex, but similarly gradient gives the mean value $E[\mathbf{x}]$ and Hessian gives the non-centered covariance $E[\mathbf{x}\mathbf{x}^T]$.

Taking derivatives with respect to ω_α in Equation (A.59), we get

$$\begin{aligned}
\frac{\partial^2 \tilde{p}}{\partial t \partial \omega_\alpha} &= \delta_{i\alpha} F_{ij} \frac{\partial \tilde{p}}{\partial \omega_j} + \omega_i F_{ij} \frac{\partial^2 \tilde{p}}{\partial \omega_j \partial \omega_\alpha} - \frac{1}{2} L_{ik} Q_{c,kk} L_{jk} \delta_{i\alpha} \omega_j \tilde{p} \\
&\quad - \frac{1}{2} L_{ik} Q_{c,kk} L_{jk} \omega_i \delta_{j\alpha} \tilde{p} - \frac{1}{2} L_{ik} Q_{c,kk} L_{jk} \omega_i \omega_j \frac{\partial \tilde{p}}{\partial \omega_\alpha} \\
&= F_{\alpha j} \frac{\partial \tilde{p}}{\partial \omega_j} + \omega_i F_{ij} \frac{\partial^2 \tilde{p}}{\partial \omega_j \partial \omega_\alpha} - \frac{1}{2} L_{\alpha k} Q_{c,kk} L_{jk} \omega_j \tilde{p} \\
&\quad - \frac{1}{2} L_{ik} Q_{c,kk} L_{\alpha k} \omega_i \tilde{p} - \frac{1}{2} L_{ik} Q_{c,kk} L_{jk} \omega_i \omega_j \frac{\partial \tilde{p}}{\partial \omega_\alpha} \\
&= F_{\alpha j} \frac{\partial \tilde{p}}{\partial \omega_j} + \omega_i F_{ij} \frac{\partial^2 \tilde{p}}{\partial \omega_j \partial \omega_\alpha} - L_{\alpha k} Q_{c,kk} L_{jk} \omega_j \tilde{p} \\
&\quad - \frac{1}{2} L_{ik} Q_{c,kk} L_{jk} \omega_i \omega_j \frac{\partial \tilde{p}}{\partial \omega_\alpha}.
\end{aligned} \tag{A.61}$$

In the limit $\omega \to 0$ all terms with multiplying ω's will vanish, which includes all but the first term on the right hand side. Thus, we get differential equation for the first moment as

$$\begin{aligned}
\frac{\partial^2 \tilde{p}}{\partial t \partial \omega_\alpha} &= F_{\alpha j} \frac{\partial \tilde{p}}{\partial \omega_j} \\
\frac{\partial}{\partial t} \frac{1}{\mathrm{i}} \frac{\partial \tilde{p}}{\partial \omega_\alpha} &= F_{\alpha j} \frac{1}{\mathrm{i}} \frac{\partial \tilde{p}}{\partial \omega_j} \\
\dot{m}_\alpha &= F_{\alpha j} m_j,
\end{aligned} \tag{A.62}$$

where we have used notation

$$E[x_\alpha] = m_\alpha. \tag{A.63}$$

If we write the Equation (A.62) in vector form, we get

$$\dot{\mathbf{m}} = \mathbf{F}\mathbf{m}. \tag{A.64}$$

Taking derivative of (A.61) with respect to ω_β, we get

$$\begin{aligned}\frac{\partial^3 \tilde{p}}{\partial t \partial \omega_\alpha \partial \omega_\beta} &= F_{\alpha j} \frac{\partial^2 \tilde{p}}{\partial \omega_j \partial \omega_\beta} + \delta_{i\beta} F_{ij} \frac{\partial^2 \tilde{p}}{\partial \omega_j \partial \omega_\alpha} - \omega_i F_{ij} \frac{\partial^3 \tilde{p}}{\partial \omega_j \partial \omega_\alpha \partial \omega_\beta} \\ &\quad - L_{\alpha k} Q_{c,kk} L_{jk} \delta_{j\beta} \tilde{p} - L_{\alpha k} Q_{c,kk} L_{jk} \omega_j \frac{\partial \tilde{p}}{\partial \omega_\beta} + o(\omega) \\ &= F_{\alpha j} \frac{\partial^2 \tilde{p}}{\partial \omega_j \partial \omega_\beta} + F_{\beta j} \frac{\partial^2 \tilde{p}}{\partial \omega_j \partial \omega_\alpha} - L_{\alpha k} Q_{c,kk} L_{\beta k} \tilde{p} + o(\omega).\end{aligned}$$

In the limit $\omega \to 0$ we get

$$\begin{aligned}\frac{\partial^3 \tilde{p}}{\partial t \partial \omega_\alpha \partial \omega_\beta} &= F_{\alpha j} \frac{\partial^2 \tilde{p}}{\partial \omega_j \partial \omega_\beta} + F_{\beta j} \frac{\partial^2 \tilde{p}}{\partial \omega_j \partial \omega_\alpha} - L_{\alpha k} Q_{c,kk} L_{\beta k} \tilde{p} \\ \frac{1}{i^2} \frac{\partial^3 \tilde{p}}{\partial t \partial \omega_\alpha \partial \omega_\beta} &= F_{\alpha j} \frac{1}{i^2} \frac{\partial^2 \tilde{p}}{\partial \omega_j \partial \omega_\beta} + F_{\beta j} \frac{1}{i^2} \frac{\partial^2 \tilde{p}}{\partial \omega_j \partial \omega_\alpha} - \frac{1}{i^2} L_{\alpha k} Q_{c,kk} L_{\beta k} \tilde{p} \\ \dot{S}_{\alpha\beta} &= F_{\alpha j} S_{j\beta} + S_{\alpha j} F_{\beta j} + L_{\alpha k} Q_{c,kk} L_{\beta k}.\end{aligned} \tag{A.65}$$

If we write the Equation (A.65) for covariance in vector form, we get

$$\dot{\mathbf{S}} = \mathbf{F}\mathbf{S} + \mathbf{S}\mathbf{F}^T + \mathbf{L}\mathbf{Q}_c\mathbf{L}^T. \tag{A.66}$$

The relationship between non-center covariance

$$\mathbf{S} = E[\mathbf{x}\mathbf{x}^T] \tag{A.67}$$

and actual covariance

$$\mathbf{P} = E[(\mathbf{x} - \mathbf{m})(\mathbf{x} - \mathbf{m})^T] \tag{A.68}$$

is

$$\mathbf{S} = \mathbf{P} + \mathbf{m}\mathbf{m}^T. \tag{A.69}$$

Taking derivatives with respect to time and taking into account that $\dot{\mathbf{m}} = \mathbf{F}\mathbf{m}$, we get

$$\dot{\mathbf{S}} = \dot{\mathbf{P}} + \dot{\mathbf{m}}\mathbf{m}^T + \mathbf{m}\dot{\mathbf{m}}^T = \dot{\mathbf{P}} + \mathbf{F}\mathbf{m}\mathbf{m}^T + \mathbf{m}\mathbf{m}^T\mathbf{F}^T. \tag{A.70}$$

Inserting this into Equation (A.66) gives

$$\begin{aligned}
\dot{\mathbf{S}} &= \mathbf{F}\mathbf{S} + \mathbf{S}\mathbf{F}^T + \mathbf{L}\mathbf{Q}_c\mathbf{L}^T \\
\dot{\mathbf{P}} + \mathbf{F}\mathbf{m}\mathbf{m}^T + \mathbf{m}\mathbf{m}^T\mathbf{F}^T &= \mathbf{F}(\mathbf{P} + \mathbf{m}\mathbf{m}^T) + (\mathbf{P} + \mathbf{m}\mathbf{m}^T)\mathbf{F}^T + \mathbf{L}\mathbf{Q}_c\mathbf{L}^T \\
\dot{\mathbf{P}} &= \mathbf{F}\mathbf{P} + \mathbf{P}\mathbf{F}^T + \mathbf{L}\mathbf{Q}_c\mathbf{L}^T. \qquad \text{(A.71)}
\end{aligned}$$

Thus, in this case the actual (centralized) covariance **P** has the same differential equation as non-centralized covariance **S**. From this derivation we get the general mean and covariance propagation equations

$$\dot{\mathbf{m}} = \mathbf{F}(t)\mathbf{m} \qquad \text{(A.72)}$$

$$\dot{\mathbf{P}} = \mathbf{F}(t)\mathbf{P} + \mathbf{P}\mathbf{F}^T(t) + \mathbf{L}(t)\mathbf{Q}_c(t)\mathbf{L}^T(t). \qquad \text{(A.73)}$$

Given two first moments $(\mathbf{m}(0), \mathbf{P}(0))$ of initial distribution $p(\mathbf{x}, 0)$, the equations above define the evolution of the first two moments in time. Derivation of these same equations and derivation of more general moment propagation equations can be found in the book of Jazwinski (1970).

References

Akashi, H. and Kumamoto, H. (1977). Random sampling approach to state estimation in switching environments. *Automatica*, 13:429–434.

Bar-Shalom, Y. and Li, X.-R. (1995). *Multitarget-Multisensor Tracking: Principles and Techniques*. YBS.

Bar-Shalom, Y., Li, X.-R., and Kirubarajan, T. (2001). *Estimation with Applications to Tracking and Navigation*. Wiley Interscience.

Bernardo, J. M. and Smith, A. F. M. (1994). *Bayesian Theory*. John Wiley & Sons.

Blackman, S. and Popoli, R. (1999). *Design and Analysis of Modern Tracking Systems*. Artech House Radar Library.

Casella, G. and Robert, C. P. (1996). Rao-Blackwellisation of sampling schemes. *Biometrika*, 83(1):81–94.

Challa, S. and Bar-Shalom, Y. (2000). Nonlinear filter design using Fokker-Planck-Kolmogorov probability density evolutions. *IEEE Transactions on Aerospace and Electronic Systems*, 36(1).

Challa, S., Bar-Shalom, Y., and Krishnamurthy, V. (2000). Nonlinear filtering via generalized Edgeworth series and Gauss-Hermite quadrature. *IEEE Transactions on Signal Processing*, 48(6).

Chen, R. and Liu, J. (2000). Mixture Kalman filters. *J. R. Statist. Soc. B*, 62:493–508.

Doucet, A. (1998). On sequential simulation-based methods for Bayesian filtering. Technical report, Signal Processing Group, Department of Engineering, University of Cambridge, UK.

Doucet, A., de Freitas, N., and Gordon, N., editors (2001). *Sequential Monte Carlo Methods in Practice*. Springer.

Duane, S., Kennedy, A. D., Pendleton, B. J., and Roweth, D. (1987). Hybrid Monte Carlo. *Physics Letters B*, 195(2):216–222.

Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. R. (1995). *Bayesian Data Analysis*. Chapman & Hall.

Gilks, W., Richardson, S., and Spiegelhalter, D., editors (1996). *Markov Chain Monte Carlo in Practice*. Chapman & Hall.

Gordon, N. J., Salmond, D. J., and Smith, A. F. M. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *IEEE Proceedings on Radar and Signal Processing*, volume 140, pages 107–113.

Grewal, M. S. and Andrews, A. P. (2001). *Kalman Filtering, Theory and Practice Using MATLAB*. Wiley Interscience.

Guenther, R. B. and Lee, J. W. (1988). *Partial Differential Equations of Mathematical Physics and Integral Equations*. Dover Publications, Inc.

Gunther, J., Beard, R., Wilson, J., Oliphant, T., and Stirling, W. (1997). Fast nonlinear filtering via Galerkin's method. In *American Control Conference. American Automatic Control Counsil.*

Gustafsson, F., Gunnarson, F., Bergman, N., Forsell, U., Jansson, J., Karlsson, R., and Nordlund, P.-J. (2002). Particle filters for positioning, navigation and tracking. *IEEE Transactions on Signal Processing*.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109.

Ho, Y. C. and Lee, R. C. K. (1964). A Bayesian approach to problems in stochastic estimation and control. *IEEE Transactions on Automatic Control*, 9:333–339.

Hue, C., Le Cadre, J.-P., and Perez, P. (2000). Tracking multiple objects with particle filtering. Technical report, IRISA.

Jazwinski, A. H. (1970). *Stochastic Processes and Filtering Theory*. Academic Press.

Julier, S. and Uhlmann, J. (1995). A general method of approximating nonlinear transformations of probability distributions. Technical report, Robotics Research Group, Department of Engineering Science, University of Oxford.

Julier, S. J., Uhlmann, J. K., and Durrant-Whyte, H. F. (1995). A new approach for filtering nonlinear systems. In *Proceedings of the 1995 American Control, Conference, Seattle, Washington*, pages 1628–1632.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Transactions of the ASME, Journal of Basic Engineering*, 82:34–45.

Kalman, R. E. and Bucy, R. S. (1961). New results in linear filtering and prediction theory. *Transactions of the ASME - Journal of Basic Engineering*, 83D:95–108.

Karlsson, R. and Gustafsson, F. (2001). Monte Carlo data association for multiple target tracking. In *IEEE Target tracking: Algorithms and applications, The Netherlands*.

Kitagawa, G. (1996). Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, 5:1–25.

Kreyszig, E. (1993). *Advanced Engineering Mathematics*. John Wiley & Sons, Inc.

Liu, J. S. (2001). *Monte Carlo Strategies in Scientific Computing*. Springer Series in Statistics. Springer.

Liu, J. S. and Chen, R. (1995). Blind deconvolution via sequential imputations. *Journal of the American Statistical Association*, 90(430):567–576.

Maybeck, P. (1979). *Stochastic Models, Estimation and Control, Volume 1*. Academic Press.

Maybeck, P. (1982a). *Stochastic Models, Estimation and Control, Volume 2*. Academic Press.

Maybeck, P. (1982b). *Stochastic Models, Estimation and Control, Volume 3*. Academic Press.

Metropolis, N., Rosenbluth, A., Rosenbluth, R., Teller, A., and Teller, E. (1953). Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21(6):1087–1092.

Milton, J. S. and Arnold, J. C. (1995). *Introduction to Probability and Statistics Principles and Applications for Engineering and the Computing Sciences*. Schaum´s Solved Problems Series. McGraw-Hill, Inc.

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

Risken, H. (1989). *Fokker-Planck Equation*. Springer-Verlag.

Stengel, R. F. (1994). *Optimal Control and Estimation*. Dover Publications, Inc.

Stone, L. D., Barlow, C. A., and Corwin, T. L. (1999). *Bayesian Multiple Target Tracking*. Artech House.

Stratonovich, R. L. (1968). *Conditional Markov Processes and Their Application to the Theory of Optimal Control*. American Elsevier Publishing Company, Inc.

van der Merwe, R., de Freitas, N., Doucet, A., and Wan, E. (2001). The unscented particle filter. In *Advances in Neural Information Processing Systems 13*. MIT Press.

Wan, E. A. and van der Merwe, R. (2001). The unscented Kalman filter. In *Kalman Filtering and Neural Networks*. Wiley Interscience.

West, M. and Harrison, J. (1997). *Bayesian Forecasting and Dynamic Models*. Springer-Verlag.