A Virtual Measurement Approach for Robust Planar Tracking

SSD: ING-INF/04

Dottorando
Francesco Conte

Coordinatore del corso
Prof. Giuseppe Ferri

Tutore
Prof. Alfredo Germani

A.A. 2011/2012
“I can calculate the motion of heavenly bodies, but not the madness of people.

(“Posso misurare il moto dei corpi, ma non l’umana follia.”)

Isaac Newton
Abstract

This thesis studies the classical problem of planar tracking. It requires the real-time estimation of the position of an object which is moving onto a plane, using noisy measurements of range and direction. A tracking algorithm may be used in several scientific and technological fields, such as aerospace, bioengineering, and robotics, to cite a few. Moreover, it is considered as a challenging problem by the system identification community and often used as benchmark for checking general nonlinear filtering methods.

Because of its relevance, a great amount of literature is dedicated to planar tracking and lots of solutions have been developed using different approaches to dealt with the nonlinearity of measurements, which constitutes the main difficulty to be addressed. The classical strategy consists on applying a linear Kalman filter to the linearized system. This often causes divergence and inconsistency. As a consequence, research is moving toward methods able to avoid linearization.

The approach here proposed is based on the idea of considering, as system output, a vector of “virtual” measurements directly obtained from the actual ones. In this way, the measurement map is split into the sum of a linear time-varying transformation of the state and an uncorrelated white noise process, which is generally nongaussian. The resulting model is amenable for applying standard linear and polynomial Kalman-like algorithms, avoiding any linearization procedure of the measurement map. The proposed algorithms are checked through numerical simulations which show improvements with respect to the existing methods in terms of robustness and computational complexity.
Acknowledgements

This thesis concludes three years of my life filled by articles, formulas, blackboards, equations, inequalities, nights, travels, swimming pools, and mountains, but especially by good friends.

First I want to thank my supervisor prof. Alfredo Germani and my doctoral mate Valerio who share with me the merit of this work and also have built with me a relationship of mutual respect and friendship that goes beyond formal work.

A thank you and a smile then go to all the laboratory of the Campus Bio-Medico who has hosted me during this period. So thanks to Filippo Cacace, Giulio Iannello and Antonio Fasano, for their support and valuable advices, and to the guys, Roberto V., Valeria, Leonardo and Roberto D.

A big hug then for my friends in Rome, Alessandro, Ludovico, Giorgia, Loris, Bruno, and my brother Marco also.

Do not offend my friends, same as ever, musicians, thinkers, workers, farmers and foodies, if I close with my parents, but you know that there is no need, than to write on a sheet or not changes little, you always will be there, and I will be too, this is enough

So the last think is for my parents. As I am always saying, the good things I do and I will do are and will be a merit of them. I hope I never stop.
# Contents

Abstract iii

Acknowledgements v

Abbreviations viii

Symbols x

1 Introduction 1

1.1 Problem Formulation .......................... 2
1.2 State of the Art .............................. 3
1.3 The Virtual Measurement Approach .............. 5
1.4 Contributions ................................ 6
1.5 Thesis Outline ............................... 6

2 Dynamical Models for Target Tracking 8

2.1 The Mathematical Representation of a Moving Target for Tracking Issues . . 8
2.2 Constant Velocity Model (CV) .......................... 10
2.3 Wiener-Process Acceleration Model .............. 11
   2.3.1 White-Noise Jerk Model (WJ) .................. 11
   2.3.2 Wiener-Sequence Acceleration Model (WS) .... 11
2.4 Zero-Mean First-Order Markov Model (MM) ........ 12

3 Nonlinear Filtering Methods for 2-D Tracking 15
### Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>c.t.</td>
<td>continuous-time</td>
</tr>
<tr>
<td>d.t.</td>
<td>discrete-time</td>
</tr>
<tr>
<td>e.g.</td>
<td>exempli gratia, for example</td>
</tr>
<tr>
<td>i.e.</td>
<td>id est, that is, namely</td>
</tr>
<tr>
<td>2-D</td>
<td>2-Dimensional</td>
</tr>
<tr>
<td>2-DTP</td>
<td>2-Dimensional Tracking Problem</td>
</tr>
<tr>
<td>ACT</td>
<td>Average Computational Time</td>
</tr>
<tr>
<td>BO</td>
<td>Bearing Only</td>
</tr>
<tr>
<td>CDF</td>
<td>Cumulative Density Function</td>
</tr>
<tr>
<td>CA</td>
<td>Constant Acceleration</td>
</tr>
<tr>
<td>CV</td>
<td>Constant Velocity model</td>
</tr>
<tr>
<td>EKF</td>
<td>Extended Kalman Filter</td>
</tr>
<tr>
<td>IEKF</td>
<td>Iterated Extended Kalman Filter</td>
</tr>
<tr>
<td>IID</td>
<td>Independent and Identically Distributed</td>
</tr>
<tr>
<td>IUKF</td>
<td>Iterated Unscented Kalman Filter</td>
</tr>
<tr>
<td>KF</td>
<td>Kalman Filter</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>LVME</td>
<td>Linear Virtual Measurement Estimate</td>
</tr>
<tr>
<td>LVMKF</td>
<td>Linear Virtual Measurement Kalman Filter</td>
</tr>
<tr>
<td>MC</td>
<td>Monte Carlo</td>
</tr>
<tr>
<td>MGEKF</td>
<td>Modified Gain Kalman Filter</td>
</tr>
<tr>
<td>MM</td>
<td>zero mean first order Markov Model</td>
</tr>
<tr>
<td>MVEKF</td>
<td>Modified Version Extended Kalman Filter</td>
</tr>
<tr>
<td>PDF</td>
<td>Probability Density Function</td>
</tr>
<tr>
<td>PF</td>
<td>Particle Filter</td>
</tr>
<tr>
<td>PSD</td>
<td>Power Spectral Density</td>
</tr>
<tr>
<td>ψ-PVME</td>
<td>ψ-Polynomial Virtual Measurement Estimate</td>
</tr>
<tr>
<td>ψ-PVMKF</td>
<td>ψ-Polynomial Virtual Measurement Kalman Filter</td>
</tr>
<tr>
<td>RPE</td>
<td>Relative Position Error</td>
</tr>
<tr>
<td>UKF</td>
<td>Unscented Kalman Filter</td>
</tr>
<tr>
<td>UT</td>
<td>Unscented Transformation</td>
</tr>
<tr>
<td>WJ</td>
<td>White-noise Jerk model</td>
</tr>
<tr>
<td>WS</td>
<td>Wiener Sequence acceleration model</td>
</tr>
</tbody>
</table>
Symbols

- $\mathbb{R}$: set of real numbers
- $\mathbb{Z}$: set of integer numbers
- $j$: imaginary unit

- $z \in \mathbb{R}^n$: real vector in $\mathbb{R}^n$
- $M \in \mathbb{R}^{n \times m}$: real matrix in $\mathbb{R}^{n \times m}$
- $(z)_i = z_i$: $i$-th element of $z$
- $(M)_{i,j}$: $(i, j)$ entry or sub-matrix of $M$
- $I_n$: identity matrix in $\mathbb{R}^n$
- $0_n$: null vector in $\mathbb{R}^n$
- $0_{n \times m}$: null matrix in $\mathbb{R}^{n \times m}$
- $\text{diag}(x) = \text{diag}(x^T)$: diagonal matrix with $x_i$ as diagonal elements
- $\text{diag}(M)$: vector composed by the diagonal elements of $M$

- $\| \cdot \|$: euclidean norm in $\mathbb{R}^n$
- $\| \cdot \|_{\mathcal{H}}$: norm in the Hilbert space $\mathcal{H}$
- $\delta(\cdot)$: Dirac delta function
Symbols

$| \cdot |_2$ 2 modulo function
$[\cdot ]$ integer part or floor function

$P(A)$ probability of the event $A$
$x \sim p(x)$ random vector $x$ distributed with density $p$
$p(x)$ probability density function of $x$
$p(x|y)$ conditional probability density of $x$, with respect to $y$
$E[x]$ expectation or mean value of $x$
$E[x|y]$ conditional expectation of $x$, with respect to $y$
$E_q[x|y]$ expectation of $x$ over the probability density function $q$
$\text{var}[x]$ covariance matrix of $x$
$\text{var}_q[x]$ covariance matrix of the $x$ over the probability density function $q$
$N(\bar{x}, \Psi_x)$ normal distribution with mean value $\bar{x}$ and covariance matrix $\Psi_x$
$U([a, b))$ uniform distribution over the real interval $[a, b)$

$O(\cdot)$ computational complexity order
Per mamma e papà...
Introduction

Target tracking is a problem of fundamental importance for both technological applications and general scientific investigation. The principal aim is to estimate the kinematic state of a moving object based on noisy nonlinear measurements of range and direction, acquired from a given observation point. The technological significance comes from the natural applications to radar and sonar systems, widely used in the civil and defence fields, but also from applications on different fields such as robotics and bioengineering. From a scientific point of view, even if characterized by a simple formalization, the target tracking is considered as a challenging filtering problem, principally due to the measurements nonlinearity and the presence of inaccuracies in the mathematical description of the target motion.

A great amount of literature has been devoted to this challenging issue with the development of many different solutions. However, all the existing algorithms may suffer from pathological behaviours such as divergence or inconsistency due to the strong nonlinearities, which make the problem intrinsically infinite dimensional and prevent from defining an implementable optimal algorithm. As a consequence, new solutions able to improve the estimate accuracy and avoid pathological behaviours are of particular interest. It is also worth mentioning that the improvement of the estimate precision is not the only issue to be addressed. Any tracking algorithm is indeed usually required to be processed in real-time, together with other routines, into an integrated technological system. Therefore, the computational burden must be always considered and limited as much as possible. In this dissertation, a new solution is presented attempting to deal with the mentioned points, here summarized:

- improve the estimation accuracy;
• improve the robustness with respect to model errors;
• limit the computation burden.

In order to make clear the scientific context and the significance of the contributions given in this thesis, after the problem formulation in Section 1.1 and the description of State of the Art in Section 1.2, the new approach is introduced in Section 1.3. Finally, Section 1.4 lists the published contributions and Section 1.5 gives an outline of the thesis.

1.1 Problem Formulation

The design objective of the 2-D tracking problem (2-DTP) is the reconstruction of the kinematic state of a moving object using noisy nonlinear measurements of distance and angle from a given observation point. The stochastic description of the planar movement of an object has received a great attention in the relevant scientific literature and several mathematical models have been proposed (see Chapter 2 for details). Most of these representations have the following discrete-time stochastic linear system form:

\[ x(k+1) = Ax(k) + Fw(k), \quad k = 0, 1, \ldots, \quad x(0) = x_0, \quad (1.1) \]

with

\[ x(k) = \begin{bmatrix} p_1(k) & \dot{p}_1(k) & \ddot{p}_1(k) & p_2(k) & \dot{p}_2(k) & \ddot{p}_2(k) \end{bmatrix}^T, \quad (1.2) \]

where \((p_1(k), p_2(k))\) are the target coordinates at time \(k\), \(\{w(k)\}\) represents the dynamical stochastic forcing process, and \(A\) and \(F\) are matrices of proper dimension whose structures depend on the chosen model. It is worth noting that the object movement is stochastically represented. This makes the description intrinsically inaccurate.

The measurement process, at each time instant \(k\), is given by the noisy values \(\rho_m(k)\) and \(\theta_m(k)\) of the target polar coordinates \(\rho(k)\) and \(\theta(k)\) [78]:

\[ \rho_m(k) = \rho(k) + v_\rho(k) = \sqrt{x_1^2(k) + x_4^2(k)} + v_\rho(k), \quad (1.3) \]
\[ \tan(\theta(k)) = \tan(\theta_m(k) - v_\theta(k)) = \frac{x_4(k)}{x_1(k)}, \quad (1.4) \]

where \(v_\rho(k)\) and \(v_\theta(k)\) denote the measurement errors.

General assumptions on the state and measurement models follow:

(1.a) \(\{w(k)\}\) is a stationary zero mean sequence in \(\mathbb{R}^r\);
(1.b) the statistical moments of \(\{w(k)\}\) are known up to a given order \(2\nu\) and summarized by vectors

\[ \psi_w^{(i)} := E \left[ w^{[i]}(k) \right], \quad i = 1, 2, \ldots, 2\nu, \quad (1.5) \]

where \([i]\) stands for the the Kronecker \(i\)-th power (see Definition A.2);
(1.c) \(x_0 \sim N(\bar{x}_0, \Psi_{x_0})\);
1.2. State of the Art

As mentioned before, since the tracking measurement process is nonlinear, the state estimation is an infinite dimensional problem. Therefore, only suboptimal solutions can be used for engineering applications. The standard tool of choice is the extended Kalman filter (EKF) algorithm \cite{3, 8, 83, 101, 103, 105, 108} which is based on the first-order linearization of the nonlinear model. Unfortunately, linearization may introduce large errors in the state estimate that sometimes cause inconsistency \cite{66} or divergence of the filter \cite{20, 110}.

A first attempt to reduce the linearization error is suggested in \cite{100} where a modified gain extended Kalman filter (MGEKF) is introduced. A basic assumption in MGEKF is that the nonlinear system is modifiable \cite{114}. Unfortunately, such a condition holds in the bearing-only (BO) location but it is not useful in the tracking case (bearing and range). In \cite{52} a modified covariance EKF (MVEKF) algorithm is developed, where it is suggested to re-linearize the

\begin{align*}
(1.d) \quad & \{v_\rho(k)\} \text{ and } \{v_\theta(k)\} \text{ are characterized by known stationary zero mean distributions, } \quad \text{whose moments are indicated with } \\
& \psi^{(i)}_\rho := \mathbb{E} [v^i_\rho(k)] , \quad \psi^{(i)}_\theta := \mathbb{E} [v^i_\theta(k)] , \quad i = 1, 2 \ldots ; \\
(1.e) \quad & \text{the initial state } x_0 \text{ forms, together with the sequences } \{w(k)\}, \{v_\rho(k)\}, \text{ and } \{v_\theta(k)\}, \text{ a family of independent variables.}
\end{align*}

Hereafter, the term \textit{white} will be referred to a sequence of uncorrelated random vectors. Notice that assumption (1.e) states that the mentioned sequences are white.

The 2-DTP requires the on-line estimation of the state \(x(k)\) by using the measurements \(\rho_m(k)\) and \(\theta_m(k)\).

\[ y_1 = \rho_m \]
\[ y_2 = \theta_m \]

Figure 1.1: 2-DTP: graphical representation.
nonlinear measurement equations at the filtered state in order to reduce the linearization approximation errors. Simulation results indicate that this method exhibits the same performance as MGEKF but without the limitation of modifiable condition.

An alternative strategy to improve the performances of EKF is given by the iterated extended Kalman filter (IEKF) [8] which adds a repeated correction procedure to the standard EKF. A crucial point of this method is the choice of the number of repetitions which is strongly connected with the filter convergence.

Although the EKF (in its many forms) is a widely used filtering strategy, over thirty years of experience with it has led to a general consensus within the tracking and control community that it is difficult to implement, difficult to tune, and only reliable for systems which are almost linear on the time scale of the update intervals [66].

A substantial improvement is obtained by the unscented Kalman filter (UKF) [65–69], which is based on the key idea of approximating the state probability density instead of the nonlinear system equations. By this intuition a filtering algorithm able to overcome the EKF in a wide range of applications [68, 109], with comparable computational complexity, is obtained.

An iterated version of UKF (IUKF) is proposed in [114]. This extension is similar to that given by IEKF for EKF, but the introduced algorithm has better convergence properties and a consistent termination criterion. Obviously, this is paid in terms of additive computational load, which grows linearly with the state dimension. For real-time applications the computation required by extra iterative procedures (e.g. IEKF, IUKF) may be considered a significant limitation. Indeed, the simplicity of EKF and UKF is often preferred to the higher accuracy of more complex techniques.

The mentioned methods use any analytical approximation of the system model or of the state probability density. A different approach is used by numerical Monte Carlo methods, also known as particle filters (PFs) [11, 12, 23, 36, 38, 46, 50, 51, 53, 58, 59, 61, 63, 72, 80, 87, 90, 95]. They are based on point mass (or “particle”) representations of probability densities [1], which do not introduce linearization. From when the main idea was presented by the seminal paper [51], PFs have become very popular and several versions have been introduced with different proposal densities, sampling and re-sampling techniques, representation of conditional density and variance reduction methods [33]. It is well known that the estimation accuracy is improved with the increase of the number of particles, at expanse of a higher computational complexity. As largely discussed (e.g. [34]), for low dimensional problems, well designed PFs achieve optimal estimation accuracy with computational complexity roughly the same of EKF. But for high dimensional problems the computational complexity of the best PF is enormous. That is, PFs suffer form the “curse of dimensionality” in general [33].

The above listed methods compose the principal class of solutions for planar tracking. However, research on nonlinear filtering has moved toward more sophisticated techniques such as numerical solutions of the Fokker-Plank equations [21, 88, 97], exact nonlinear recursive filters [10, 31, 32], and batch or nonlinear nonrecursive filters [47, 93, 102]. Most of these methods significantly improve the performance of EKF but with a larger computational complexity. Moreover, implementations are not always easy and require high performance computational resources.

This dissertation focuses on the most popular and used methods. In particular, EKF, UKF and PFs is described with more details in Chapter 2 and the new technique is compared with these solutions in Chapter 5.
1.3 The Virtual Measurement Approach

In this work, a new philosophy for solving the 2-DTP is introduced. The main idea is to avoid linearization by using the concept of virtual measurement map, formalized by the following definition.

**Definition 1.1.** Let

\[
x(k+1) = Ax(k) + Bu(k) + Fw(k),
\]

\[
y(k) = h(x(k), u(k), v(k)),
\]

(1.6)
(1.7)

describe a dynamical system whose state \(x(k)\) evolves in \(\mathbb{R}^n\), \(y(k) \in \mathbb{R}^q\) is the available measurement vector, and \(u(k) \in \mathbb{R}^p, w(k) \in \mathbb{R}^r\), and \(v(k) \in \mathbb{R}^s\) are the deterministic input, the state noise, and the output measurement error vectors, respectively. Such a system is said to admit a virtual measurement map if there exist a completely observed sequence \(\{y_v(k)\}\), named virtual output, a \(q \times n\) matrix \(C(k)\), and a random white sequence \(\{n(k)\}\), not necessary Gaussian, endowed with all moments up to a certain order \(2\nu\), and uncorrelated with the state sequence \(\{x(k)\}\), such that, for any \(k \geq 0\), one has:

\[
y_v(k) = C(k)x(k) + n(k).
\]

Definition 1.1 roughly says that a system admit a virtual measurement map if there exists a known transformation of the measurements which is linearly related to the system state and preserves the whiteness of the transformed noise. When such a condition holds true, the considered filtering problem can be readily solved by applying standard Kalman Filter (KF) to the transformed system. In this way, no linearization needs to be introduced at expense of losing the gaussianity of the measurement noise, even if the original one is Gaussian. It is worth stressing that the algorithm remains the best linear estimator with respect to the defined virtual measurement. It is indeed well known that KF is the linear least square estimator with respect to the considered model.

As far as the nongaussianity of the transformed noise is concerned, the use of a polynomial Kalman-like algorithm [18, 19] allows improving the performances of the linear one. Of course, in this case, complexity could become less negligible.

Notice that such an approach has a general value: each discrete-time dynamical system endowed with a virtual measurement map admits to be processed by standard KF to obtain the minimum variance state estimate. Moreover, Definition 1.1 can be reduced to the deterministic case and a similar method can be used to solve state observation problems (e.g. [26]).

In this thesis, the mentioned linear and polynomial algorithms are developed for solving the 2-DTP. Model (1.1), (1.3)-(1.4) has indeed the form (1.6)-(1.7) and can be proved to admit a virtual measurement map, as shown in Chapter 4. In particular, range and angle measurements becomes time-varying parameters of the output matrix and of the “virtual” output vector, which is linearly related to the system state. The effectiveness of the new approach is validated through several simulation results provided in Chapter 5, where the two algorithms exhibit improvements to the best existing methods in terms of robustness with respect to model errors, with a comparable computational load.
1.4 Contributions

A list of the contributions related to the main subject of this thesis follows.

In [28]


the approach is briefly introduced with preliminary results. The performances of the algorithm are then checked in a non-Gaussian setting by [27]


where the focus is moved to the analysis of the algorithm robustness with respect to a realistic nongaussian behaviour of the target movement. Finally, a complete mathematical description of the proposed approach is given in [29]

F. Conte, V. Cusimano and A. Germani. Robust Planar Tracking via a Virtual Measurement Approach, European Journal of Control, 2013, Accepted for publication.

together with a deep analysis of the algorithm performances in Gaussian and nongaussian settings. A particular attention is given to robustness with respect to model errors and computational burden. A comparison with the existing nonlinear filtering methods (EKF, UKF, PF) is also used to validate the new technique. Most of the material provided in this Ph.D. thesis is extracted from [29].

1.5 Thesis Outline

This thesis is organized in the following way:

- **Chapter 1 Introduction** (this chapter) Introduces the research problem and lists the published contributions.

- **Chapter 2 Dynamical Models for Target Tracking** Describes the main models for the target motion.

- **Chapter 3 Nonlinear Filtering Methods for 2-D Tracking** Briefly introduces the principal existing algorithms for nonlinear/nongaussian filtering, usually used for tracking issues.

- **Chapter 4 LVMKF and \( \nu \)-PVMKF for the 2-DTP** Proves the tracking model to be endowed with a virtual measurement map and introduces the basic linear filtering algorithm and the polynomial extension.

- **Chapter 5 Results** Presents the simulation results for several kinematics models given in terms of estimation accuracy and computational load.
• **Chapter 6 Conclusions** Concludes the thesis and gives ideas for future expansions.

• **Appendix A Kronecker Algebra** Provides some key mathematical tools, widely used in the text.
Dynamical Models for Target Tracking

The key to successful target tracking lies in the effective extraction of useful information about the target state from observations. A good model of the target will certainly facilitate this information extraction to a great extent. Most tracking algorithms are model-based because knowledge of target motion is available and a good model-based tracking algorithm will greatly outperform any model-free tracking algorithm if the underlying model turns out to be a good one [79]. As a consequence, a crucial task in the design of a tracking algorithm is the selection of a good dynamical model by the analysis of the relation between the tracking performance and the used representations.

Various models of target motion have been developed over the past four decades. There are several books and surveys [4–8, 13–15, 42, 45, 48, 71, 75, 81, 82, 91, 92, 98] that give a presentation of this great amount of mathematical representations. In this chapter, four of the mostly used models are considered and described coherently with the formulation given in Chapter 1.

2.1 The Mathematical Representation of a Moving Target for Tracking Issues

Although a moving object is almost never a really point in space, a target is usually treated as a point object without a shape in tracking. A target model has to describe the evolution of the target state with respect to time. A possible description can be expressed by the state-space
2.1. The Mathematical Representation of a Moving Target for Tracking Issues

**continuous-time (c.t.) system form**

\[
\dot{x}_c(t) = f_c(x_c(t), u_c(t), w_c(t), t), \quad t \in [0, \infty) \subset \mathbb{R}, \quad x_c(0) = x_0,
\]

where \(x_c(t)\) is the target state, \(u_c(t)\) is the control input, and \(w_c(t)\) is any stochastic forcing process which carries all the uncertainties of the model. Function \(f_c\) is possibly nonlinear and/or time-varying. Since measurements are usually sampled signals, in order to make easily tractable the filtering problem, **discrete-time (d.t.) models are considered:**

\[
x(k + 1) = f(x(k), u(k), w(k), k), \quad k = 0, 1, 2, \ldots, \quad x(0) = x_0, \tag{2.1}
\]

where \(x(k), u(k), w(k),\) and \(f\) are the d.t. counterparts of state, control, stochastic forcing process, and dynamic function, respectively. Such a representation can be directly defined in d.t. or obtained by **discretizing (or sampling)** a c.t. model. In the former case, the model is referred to as **direct d.t model**; in the latter case, it is called **equivalent d.t model**. An equivalent d.t model is usually more systematic and consistent, and is in many cases preferable to the corresponding direct d.t. model. For example, target motion should not depend on how and when samples are taken [79].

The precision and tractability of a model depend on the different possible choices that can be made on characterizing its mathematical components.

A first decision regards the form of function \(f\). Although the world is nonlinear, nonlinearities are hard to be treated. As consequence, since the characterization of the target always is an approximation, it is usually preferred to build a more accurate linear model with a bounded number of parameters, instead of using a more sophisticated nonlinear representation. Moreover, the dependence of \(f\) on time might be sometimes difficult to be treated and not really useful for a suitable modelling.

As far as the control input \(u\) is concerned, it is usually not known in tracking. A crucial aspect of tracking is indeed the lack of any real-time communication from the target. This is substantial since that is what really makes hard the tracking issue and, in particular, the modelling of target motion. The actually nonrandom control input is tough approximated as a random process of certain properties. More precisely, it is neglected and completely modelled by the random process \(w\).

Therefore, the usual tracking model has the form of the linear time-invariant version of (2.1):

\[
x(k + 1) = Ax(k) + Fw(k), \quad x(0) = x_0, \tag{2.2}
\]

which corresponds to (1.1).

The final and fundamental task for getting the model is the characterization of the forcing process \(w\). In fact, the description of the target behaviour completely depends on the physical meaning and the stochastic features defined for this random process. Roughly speaking, different processes yield different models.

The physical meaning of \(w\) also determines the composition of the state vector. It can be only composed by position and velocity, or include the acceleration also. Anyway, all models here described are expressed in the d.t. form (2.2), with the full state vector

\[
x(k) = \begin{bmatrix}
p_1(k) & \dot{p}_1(k) & \ddot{p}_1(k) & p_2(k) & \dot{p}_2(k) & \ddot{p}_2(k)
\end{bmatrix}^T,
\]
2.2. Constant Velocity Model (CV)

with \((p_1(k), p_2(k))\) being the d.t. planar coordinates of target, as defined in (1.2).

A final note: most target maneuvers are coupled across different coordinates. For simplicity, however, many models developed assume that this coordinate coupling is weak and can be neglected. That is particularly the case for those that model the control input \(u\) as a random process [79]. Hence, the actual form of matrices \(A\) and \(F\) is

\[
A = \begin{bmatrix} \bar{A} & 0 \\ 0 & \bar{A} \end{bmatrix}, \quad F = \begin{bmatrix} \bar{F} & 0 \\ 0 & \bar{F} \end{bmatrix}.
\]

This means that only one coordinate direction is required to be considered. For convenience, the following notation will be used: \(p_c(t)\) and \(p(k)\) indicates the one-direction c.t. coordinate (one among \(p_1\) and \(p_2\)) and its d.t counterpart, respectively, and

\[
\bar{x}(k) := \begin{bmatrix} p(k) \\ \dot{p}(k) \\ \ddot{p}(k) \end{bmatrix}
\]

is the one-direction d.t. state vector. Hence, the model to be defined is

\[
\bar{x}(k + 1) = \bar{A}\bar{x}(k) + \bar{F}w(k).
\]

2.2 Constant Velocity Model (CV)

The simplest target to be considered for modelling is a nonmaneuvering object. This can be easily represented by setting \(\ddot{p}_c(t) = 0\). In practice, this ideal equation is usually modified as \(\ddot{p}_c(t) = a_c(t)\), where \(a_c(t)\) is a zero mean white-noise process with power spectral density (PSD) equal to \(\sigma_a^1\). This has an effect on \(p_c(t)\) that accounts for unpredictable modelling errors due to turbulences or other stochastic phenomena. The PSD \(\sigma_a^2\) regulates the amount of such an effect: when it is sufficiently large, velocity is “allowed” to significantly change, and viceversa. Because of these hypothesis, the model is referred to as constant velocity model (CV) or, sometimes, nearly constant velocity when \(\sigma_a\) is small, and white-noise acceleration model when \(\sigma_a\) is sufficiently large [8].

The c.t. CV model has the form

\[
\begin{bmatrix} \dot{p}_c(t) \\ \ddot{p}_c(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} p_c(t) \\ \dot{p}_c(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \sigma_a \end{bmatrix} w_c(t)
\]

where \(w_c(t)\) is a zero mean white-noise process with unitary PSD.

The discretization of a linear stochastic model can be carried out in a closed form [8]. To fully describe the derivation of models is not of interest in this dissertation. Therefore, the discretization procedure is not detailed.

---

1 A stationary process \(x(t) \in \mathbb{R}, t \in \mathbb{R}\) is a zero mean white-noise process iff:

- \(E[x(t)] = 0, \forall t \in \mathbb{R},\)
- \(E[x(t)x^T(t')] = 0, \forall t, t' \in \mathbb{R} \text{ s.t. } t \neq t',\)
- \(E[x(t)x^T(t)] = \sigma_x^2 I_n, \forall t \in \mathbb{R},\)

where \(\sigma_x^2\) is the PSD of \(x(t)\).
By indicating the sampling period with $\Delta$, the equivalent d.t. model is given by (2.2), where:

$$\bar{A} = \begin{bmatrix} 1 & \Delta & \Delta^2/2 \\ 0 & 1 & \Delta \\ 0 & 0 & 0 \end{bmatrix}, \quad \bar{F} = \begin{bmatrix} 0 \\ 0 \\ \sigma_a \end{bmatrix},$$

(2.3)

and $\{w(k)\}$ is a standard white sequence, usually supposed to be Gaussian.

2.3 Wiener-Process Acceleration Model

The second simplest model is the so-called white-noise acceleration model. It assumes that the target acceleration is a Wiener process, or more generally and precisely, the acceleration is a process with independent increments, which is not necessary a Wiener process. Sometimes it is referred to as constant-acceleration (CA) model or nearly-constant-acceleration model [79].

Two different version are commonly used: the white-noise Jerk model (WJ) and the Wiener-sequence acceleration model (WS).

2.3.1 White-Noise Jerk Model (WJ)

This model assumes that the acceleration derivative $\dot{a}_c(t)$, called “jerk”, is a white-noise process with PSD $\sigma_a^2/\tau^3$, where $\tau$ is used to regulate the supposed maneuver capability of the target. A large $\tau$ reduces the PSD, and so the possible changes of acceleration; a small $\tau$ has the opposite effect.

By these assumptions the c.t. model has the form:

$$\begin{bmatrix} \dot{p}_c(t) \\ \dot{\dot{p}}_c(t) \\ \dot{a}_c(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_c(t) \\ \dot{p}_c(t) \\ a_c(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \sqrt{2/\tau} \sigma_a \end{bmatrix} w_c(t)$$

and $w_c(t)$ is a zero mean white-noise process with unitary PSD. Its d.t. equivalent is given by (2.2), where:

$$\bar{A} = \begin{bmatrix} 1 & \Delta & \Delta^2/2 \\ 0 & 1 & \Delta \\ 0 & 0 & 1 \end{bmatrix}, \quad \bar{F} = \sigma_a \sqrt{2/\tau} \begin{bmatrix} \Delta^5/20 & \Delta^4/8 & \Delta^3/6 \\ \Delta^4/8 & \Delta^3/3 & \Delta^2/2 \\ \Delta^3/6 & \Delta^2/2 & \Delta \end{bmatrix}^{1/2},$$

(2.4)

and $\{w(k)\}$ is a standard white sequence. It is often supposed to be Gaussian, which corresponds to assume $a_c(t)$ as a Wiener process.

2.3.2 Wiener-Sequence Acceleration Model (WS)

This model assumes that the acceleration increment is a white-noise process. An acceleration increment over a time period is the integral of the jerk over such a period. In this case, the

---

$^2$The term standard means that the variance of the sequence is equal to one.

$^3$A process with independent increments is the Wiener integral of a white-noise process. When the latter is Gaussian, the integral returns a Wiener process.
model is most conveniently expressed in d.t. directly, with the form 2.2, with:

\[
\tilde{A} = \begin{bmatrix}
1 & \Delta & \Delta^2/2 \\
0 & 1 & \Delta \\
0 & 0 & 1
\end{bmatrix}, \quad \tilde{F} = \sigma_a \begin{bmatrix}
\Delta^2/2 \\
\Delta \\
1
\end{bmatrix},
\]

and \{w(k)\} being a standard white sequence. As for the WJ model, Gaussianity of noise is usually assumed.

### 2.4 Zero-Mean First-Order Markov Model (MM)

The above models are simple but crude. This is principally due to the use of white-noise for modelling the forcing processes. In fact, white-noise is isolated in time since its value at one time is uncoupled with any other time. During a maneuver this does not hold. The forcing process at any time may be indeed strongly dependent of what has happened in the immediate preceding time. Consequently, it is more appropriate to consider a Markov process, which is local in time because its value at one time depends on values at other times only through its nearest neighbours.

The zero-mean first-order Markov model (MM), also known as Singer model [98], assumes that the target acceleration \( a_c(t) \) is a zero mean first-order Markov stationary random process with autocorrelation

\[
r(\delta) := E[a_c(t + \delta)a_c(t)] = \sigma_a e^{-|\delta|/\tau}, \quad \tau > 0
\]

or, equivalently power spectrum

\[
R(s) = \frac{-2\tau\sigma_a^2}{s^2\tau^2 - 1}.
\]

Such a process can be obtained as solution of the stochastic system\(^4\)

\[
\dot{a}_c = -\frac{1}{\tau}a_c(t) + \sqrt{\frac{2}{\tau}}\sigma_a w_c(t),
\]

where \( w_c(t) \) is a zero mean white-noise process with constant unitary PSD. The c.t. state-space representation of the MM model is:

\[
\begin{bmatrix}
\dot{p}_c(t) \\
\ddot{p}_c(t) \\
\dot{a}_c(t)
\end{bmatrix} = \begin{bmatrix}
0 & 1 & 0 \\
0 & 0 & 1 \\
0 & 0 & -\frac{1}{\tau}
\end{bmatrix} \begin{bmatrix}
p_c(t) \\
\dot{p}_c(t) \\
a_c(t)
\end{bmatrix} + \begin{bmatrix}
0 \\
0 \\
\sqrt{\frac{2}{\tau}}\sigma_a
\end{bmatrix} w_c(t).
\]

\(^4\) A stationary Markov process with a rational power spectrum (as is the case here) is equivalent to the state of an asymptotically stable linear time-invariant system excited by strictly white noise: every such process can be represented as the state of such a system and the state of such a system is such a Markov process [79].
2.4. Zero-Mean First-Order Markov Model (MM)

The d.t. equivalent model is given by (2.2), where:

\[
\bar{A} = \begin{bmatrix}
1 & \Delta & \tau \left( \Delta - \tau + \tau e^{-\Delta/\tau} \right) \\
0 & 1 & \tau \left( 1 - e^{-\Delta/\tau} \right) \\
0 & 0 & e^{-\Delta/\tau}
\end{bmatrix},
\]

(2.6)

\(\bar{F}\) is the square root matrix of the covariance matrix given in e.g. [8, 14, 98], and \(\{w(k)\}\) is a standard white sequence.

The success of the MM relies on an accurate determination of the parameters \(\sigma_a\) and \(\tau\) [79, 99]. Parameter \(\tau\) is the maneuver time constant and thus depends on how long the maneuver lasts. For example, considering an aircraft, \(\tau \approx 60\) s is suitable for a lazy turn, and \(10 - 20\) s are possible values for representing an evasive maneuver, as suggested in [98]. Parameter \(\sigma_a = E[a_c^2(t)]\) is the “instantaneous variance” of the acceleration. In [98] the ternary-uniform mixture uniform distribution in Figure 2.1 was proposed to model the acceleration: the target may move without acceleration with probability \(P_0\), accelerate or decelerate at a maximum rate \(a_m\) with equal probability \(P_m\), or accelerate or decelerate at a rate uniformly distributed over \((-a_m, a_m)\). It turns out that

\[
\sigma_a^2 = \frac{a_m^2}{3} (1 + 4P_m - P_0)
\]

where \(a_m\), \(P_0\), and \(P_m\) are design parameters. It is worth emphasizing that MM is a maneuver model and thus \(P_0\) should be the probability of having a zero acceleration during a maneuver, rather than probability of a nonmaneuvering motion. Note also that this ternary-uniform mixture distribution of acceleration can obviously be used for other maneuver models and it is used here only to determine \(\sigma_a\).

It is clear that in the limit:

1. as the maneuver time constant \(\tau\) increases MM reduces to CA models. This relationship makes sense since the deterministic part of the acceleration in MM becomes constant in the limit as \(\tau\) increases;

2. on the other hand, as the maneuver time constant \(\tau\) decreases MM reduces CV. In this case, the acceleration becomes white-noise.
Consequently, for a choice of $0 < \tau \Delta < \infty$, MM corresponds to a motion in between of (nearly) constant velocity and (nearly) constant acceleration. It should thus be clear that MM has a wider coverage than the CV and CA models.

Many other models have been proposed (see, e.g. [15, 45, 75, 79, 91, 92]) which are equivalent to or are simple variants of MM. It has been applied in [81] for angular acceleration as well as linear acceleration. It is interesting that the same model, except that $\tau$ is data-dependent and time-varying, was obtained approximately in [40] for the angular velocity components (pitch and yaw) of the relative motion between the target and seeker based on well-known relationships in classical mechanics.

The Singer acceleration model used for MM is a popular model for target maneuvers. It was the first model that characterizes the unknown target acceleration as a time-correlated (i.e. colored) stochastic process, and has served as a basis for further development of many effective models, such as the mean-adaptive acceleration model and the asymmetrically distributed normal acceleration model, or models which describe a particular maneuvering scenario, such as the Markov models for oscillatory targets, the Markov acceleration model for constant turns (CT), and the semi-Markov jump process models [79].

However, the presented models are more general and able to describe a wider class of target behaviours. More precisely, the CV is the simplest model but it is only suitable for representing constant velocity targets; WJ and WS (CA models) are more appropriate for describing maneuvering targets but not really suitable for regular constant velocity movements; finally the MM has a wider coverage than the CV and CA models.

In general, no model is able to fully describe any target behaviour. Thus errors are unavoidable. The tracking strategy is hence required to be robust with respect to such errors, which become more significant when the a wrong model is used. The results’ Chapter 5 discusses this aspect of tracking using all the above detailed models.
Nonlinear Filtering Methods for 2-D Tracking

Nonlinear filtering is a challenging problem of system identification. It has a wide field of applications, beyond the planar tracking. To cite a few, nonlinear filtering is applied to electronics [112], robotics [25, 30, 106], navigation [8], geolocalization [60], aerospace [70, 115], biology [24], medical sciences [43], finance [111], meteorology [57].

The aim of this chapter is to extend the description of the State of the Art on nonlinear filtering given in Section 1.2, by providing some mathematical details about EKF, UKF, and PFs. These three algorithms have been selected among the plethora of existing nonlinear filters since they have been widely used to solve tracking issues and, in general, are good representatives of the relevant literature.

The chapter is organized in the following way: Section 3.1 introduces the nonlinear filtering problem and rewrites the 2-DTP in such a general form; Section 3.2 states the concept of optimality for a filtering solution and introduces the KF algorithm; finally, Sections 3.3, 3.4, and 3.5 describes EKF, UKF, and PFs, respectively, including also their suitable versions for planar tracking.
3.1 The Nonlinear Filtering Problem

The nonlinear filtering problem requires the recursive estimation of the state $x(k) \in \mathbb{R}^n$ of a nonlinear stochastic discrete-time system having the form:

$$
x(k + 1) = f(x(k), u(k), w(k), k), \quad x(0) = x_0 \tag{3.1}
$$

$$
y(k) = h(x(k), u(k), v(k), k), \tag{3.2}
$$

where $k = 0, 1, \ldots$, $u(k) \in \mathbb{R}^p$ is the deterministic control input, $y(k) \in \mathbb{R}^q$ is the available measurement vector, $\{w(k) \in \mathbb{R}^r\}$ is the state noise sequence, $\{v(k) \in \mathbb{R}^s\}$ is the measurement error sequence, and $f : \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^r \rightarrow \mathbb{R}^n$ and $h : \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^s \rightarrow \mathbb{R}^q$ are possibly nonlinear functions. Moreover, the following properties are usually assumed:

1. \{(w(k))\} and \{(v(k))\} are independent zero mean white sequences of random vectors with the covariance matrices $\Psi_w(k) \in \mathbb{R}^{n \times n}$ and $\Psi_v(k) \in \mathbb{R}^{q \times q}$, respectively;
2. the initial state $x_0 \in \mathbb{R}^n$ is a random vector with the mean value $\bar{x}_0$ and covariance matrix $\Psi_{x_0} \in \mathbb{R}^{n \times n}$, independent of $\{w(k)\}$ and $\{v(k)\}$.

The estimation is required to be carried out starting from the initial estimate $\bar{x}_0$ and using the measurement sequence $\{y(\tau), \tau = 0, 1, \ldots, k\}$, which can be collected in the $(k + 1)q$ vector

$$
Y_k := \begin{bmatrix} y(0) \\ y(1) \\ \vdots \\ y(k) \end{bmatrix}.
$$

Since this thesis is dedicated to the 2-DTP, introduced in Section 1.1, model (1.1), (1.3)-(1.4) is rewritten in the form (3.1)-(3.2):

$$
x(k + 1) = f(x(k), u(k), w(k), k) = Ax(k) + Fw(k), \tag{3.3}
$$

$$
y(k) = h(x(k), u(k), v(k), k) = \begin{bmatrix} \sqrt{x_1^2(k) + x_4^2(k)} + v_1(k) \\ \arctan\left(\frac{x_4(k)}{x_1(k)}\right) + v_2(k) \end{bmatrix}, \tag{3.4}
$$

where:

- $x(k) \in \mathbb{R}^6$ is the 2-D target configuration (1.2) (position, velocity, and acceleration);
- $A \in \mathbb{R}^{6 \times 6}$ and $F \in \mathbb{R}^{6 \times r}$ are the target dynamical and forcing matrices, respectively, which depend on the chosen model (see Chapter 2);
- $\{w(k)\}$ is the stochastic forcing process evolving in $\mathbb{R}^r$;
- $v(k) = [v_1(k) \ v_2(k)]^T \in \mathbb{R}^2$ is the measurement error vector, i.e. $v_1(k) := v_\rho(k)$ and $v_2(k) := v_\theta(k)$.
Moreover, in virtue of properties (1.a)-(1.e), \( \{ w(k) \}, \{ v(k) \} \), together with the initial state \( x_0 \), satisfy properties (3.a) and (3.b), and their relative covariance matrices are given by

\[
\Psi_w(k) = \Psi_w = s \! t^{-1} \left( \psi^{(2)}_w \right), \quad \Psi_v(k) = \Psi_v = \begin{bmatrix}
\psi_v^{(2)} & 0 \\
0 & \psi_\theta^{(2)}
\end{bmatrix},
\]

respectively. Operator \( s \! t^{-1}(\cdot) \) computes the inverse stack, defined in Definition A.3.

## 3.2 Optimal Filtering

From a probabilistic perspective, any statistical estimate of \( x(k) \) will be some function of the conditional density:

\[
p(x(k)|Y_k)
\]

since it represents all the information which the measurement of the random vectors belonging to \( Y_k \) has conveyed about the random vector \( x(k) \). This statistical estimate is denoted by \( \hat{x}(k) \). Suppose now that \( \hat{x}(k) \) is given as a fixed function of the random vectors in \( Y_k \). Then \( \hat{x}(k) \) itself is a random vector and its actual value is known whenever the actual value of \( Y_k \) is known. In general, the actual value of \( \hat{x}(k) \) is different from the unknown value of \( x(k) \) [70]. To arrive to a rational way of determining \( \hat{x}(k) \), it is natural to define an optimality criterion. This can be done in different ways. Next, the two most used approaches are presented. It will be clear that both of them provide the same theoretical result about the optimal solution. However, it is worth noting that different approaches may yield vary different suboptimal solutions (e.g. EKF, UKF, PFs).

### 3.2.1 Minimum Variance Estimate

Estimate optimality is usually defined by assigning a penalty or loss for incorrect estimates. Clearly, the loss should be a positive, nondecreasing function of the estimation error

\[
\hat{e}(k) := x(k) - \hat{x}(k).
\]

The common choice for such a function is

\[
L(\hat{e}(k)) = E \left[ \| \hat{e}(k) \|^2 \right],
\]

\( \| \cdot \| \) being the euclidean norm in \( \mathbb{R}^n \). This choice leads up to the so called minimum variance estimate which is the random vector \( \hat{x}(k) \) that minimizes \( L(\hat{e}(k)) \). It is well known that the minimum variance estimate is the state conditional expectation, i.e.

\[
\hat{x}(k) = E \left[ x(k)|Y_k \right].
\]

Moreover, \( E [\hat{x}(k)] = E [x(k)] \), i.e. \( \hat{x}(k) \) is unbiased.

It is useful to show that the minimum variance estimate can be also argued by using a geometrical approach, as done in [18]. Let \((\Omega, \mathcal{F}, P)\) be a probability space. For any given sub \( \sigma \)-algebra \( \mathcal{G} \) of \( \mathcal{F} \), denote by \( L^2(\mathcal{G}, n) \) the Hilbert space of the \( n \)-dimensional, \( \mathcal{G} \)-measurable,
3.2. Optimal Filtering

random vectors with finite second moment as

\[ L^2(G, n) := \left\{ x : \Omega \rightarrow \mathbb{R}^n, G\text{-measurable}, \int_\Omega \| x(\omega) \|^2 dP(\omega) < \infty \right\}, \]

where \( \int_\Omega \cdot dP(\omega) \) indicates the Lebesgue integral. This Hilbert space is endowed with the internal product

\[ [x, z] := \int_\Omega x^T(\omega)z(\omega)dP(\omega) = E[x^Tz] \]

and the induced norm

\[ \| x \|_{L^2} := \sqrt{\int_\Omega x^T(\omega)x(\omega)dP(\omega)} = \sqrt{E[\| x \|^2]}. \]

Moreover, when \( G \) is the \( \sigma \)-algebra generated by a random vector \( y : \Omega \rightarrow \mathbb{R}^m \), that is \( G = \sigma(y) \), the notation \( L^2(y, n) \) indicates \( L^2(\sigma(y), n) \). Finally, if \( B \) is a subspace of \( L^2(F, n) \), the symbol \( \Pi(x|B) \) indicates the orthogonal projection of \( x \in L^2(F, n) \) onto \( B \).

From the Hilbert projection theorem [2] follows that the minimum variance estimate of a random vector \( x \in L^2(F, n) \) with respect to a random vector \( y \) corresponds to \( \hat{x}(k) = E[x(k)|Y_k] = \Pi(x(k)|L^2(Y_k, n)) \). (3.6)

3.2.2 Bayesian Optimal Estimate

From a Bayesian perspective, to get an optimal estimate means to completely characterize the density \( p(x(k)|Y_k) \), which actually corresponds to say that \( \hat{x}(k) \) is the conditional expectation of \( x(k) \).

3.2.3 The Kalman Filter as Linear Optimal Estimator

In the general nonlinear/nongaussian case, to determine \( E[x(k)|Y_k] \) is an infinite dimensional problem. However, there exist cases that admit an optimal solution, such as the linear Gaussian one:

\[ x(k+1) = A(k)x(k) + B(k)u(k) + F(k)w(k), \quad x(0) = x_0 \]
\[ y(k) = C(k)x(k) + v(k), \]

(3.7) (3.8)

where \( A(k), B(k), F(k), \) and \( C(k) \) are real matrices of suitable dimensions and the following properties are satisfied together with (3.a) and (3.b):

(3.c) \( x_0 \sim N(\bar{x}_0, \Psi_{x_0}) \),
(3.d) \( w(k) \sim N(0, \Psi_w(k)) \),
(3.e) \( v(k) \sim N(0, \Psi_v(k)) \),

\[ \text{For the sake of simplicity, and without loss of generality, the dependence of the measurement vector } y(k) \text{ on the control input } u(k) \text{ has been omitted.} \]
3.2. Optimal Filtering

Algorithm 3.1: Kalman Filter

for all $k = 0, 1, \ldots$. Algorithm 3.1 reports the KF standard procedure, where the following notation is used:

- $\hat{x}(k|k - 1)$ is the state prediction, i.e. the estimate of the system state at time $k$, given the sequence $Y_{k-1}$;
- $\hat{x}(0|\cdot)$ is the initial estimate, assumed to be the prediction of the initial state;
- $P(k|k - 1)$ is the prediction error covariance matrix at time $k$;
- $P(0|\cdot)$ is the initial error covariance matrix;
- $\hat{x}(k|k)$ is the state estimate at time $k$;
- $P(k|k)$ is the estimate error covariance matrix at time $k$.

This notation will be used for all algorithms presented in this thesis.

The filtering procedure starts from the initial estimate $\bar{x}_0$ and the initial error covariance matrix $\Psi_{x_0}$ (line 2). The initial estimate is actually considered as prediction of the initial state $x(0)$. Then, the filtering consists of two steps: the correction or measurement update (lines 5-7), where the measurements are used to correct the prediction $\hat{x}(k|k-1)$ and obtain the estimate $\hat{x}(k|k)$, and the prediction or time update (lines 9-10), where the prediction $\hat{x}(k+1|k)$ of the state at the future time $k+1$ is computed.

The following proposition states that the KF recursively provides the optimal solution to the filtering problem for linear Gaussian systems:

**Proposition 3.1.** Consider a linear Gaussian system having the form (3.7)-(3.8) and satisfying assumptions (3.a)-(3.e). Then, KF returns the actual value $\hat{x}(k|k)$ and the error covariance
matrix \( P(k|k) \) of the minimum variance estimate \( \hat{x}(k) = E[x(k)|Y_k] \), provided the measurement sequence \( Y_k \).

Such a proposition has been proved based on different approaches. An interesting way to understand how KF obtains the optimal estimate and roughly prove Proposition 3.1 comes from the geometrical approach above described, which actually is the one used by Kalman in his famous paper [70]. Let us define

\[
y' := \begin{bmatrix} 1 \\ y \end{bmatrix},
\]

and the sub-space

\[
\mathcal{L}(y, n) := \left\{ z : \Omega \to \mathbb{R}^n \mid \exists M \in \mathbb{R}^{n \times (m+1)} \text{ s.t. } z = My' \right\} \subset L^2(y', n) \equiv L^2(y, n),
\]

which is composed by all affine transformations of the random vector \( y : \Omega \to \mathbb{R}^m \). From the Hilbert projection theorem [2], it follows that the projection \( \Pi(x|L^2(y, n)) \) is the minimum variance linear estimate of \( x \in L^2(F, n) \) with respect to \( y \). Moreover, if \( x \) and \( y \) are jointly Gaussian, the minimum variance estimate is linear, i.e.

\[
E[x|y] = \Pi(x|L^2(y, n)) = \Pi(x|\mathcal{L}(y, n)).
\]

Now, the following proposition characterizes the KF from a geometric point of view:

**Proposition 3.2.** Consider a system having the form (3.7)-(3.8) and satisfying properties (3.a) and (3.b). Then, KF returns the actual value \( \hat{x}(k|k) \) and the error covariance matrix \( P(k|k) \) of \( \Pi(x(k)|\mathcal{L}(Y_k, n)) \), provided the measurement sequence \( Y_k \).

A proof can be found in [70]. This proposition means that KF returns the state minimum variance linear estimate of a linear generally nongaussian system. Then, by taking into account (3.10), it results that the KF is able to compute the state minimum variance estimate of a linear Gaussian system, which is what Proposition 3.1 states. In other words, KF is a suboptimal linear estimator for linear nongaussian systems, and optimal for linear Gaussian systems.

Finally, it is useful to recognize KF also as the optimal Bayesian estimator for linear Gaussian systems. It is indeed well known that a Gaussian variable remains Gaussian if managed by an affine transformation. Moreover, by definition of conditional density, since \( x(k) \) and \( Y_k \) are jointly Gaussian, \( p(x(k)|Y_k) \) also is Gaussian. It can be proved that \( \hat{x}(k|k) \) and \( P(k|k) \), which are recursively provided by KF, actually correspond to the mean value and covariance matrix of \( p(x(k)|Y_k) \) [1], which, because of gaussianity, completely define this conditional density, i.e.

\[
p(x(k)|Y_k) = N(\hat{x}(k|k), P(k|k)).
\]

### 3.3 Extended Kalman Filter

The EKF is “the” classical filter for nonlinear systems. Despite it suffers form significant drawbacks, for many time, it has been considered the standard tool of choice for real tracking

\footnote{For simplicity, in this thesis the term linear often means affine.}
3.3. Extended Kalman Filter

applications. Moreover, from a scientific point of view, it is considered as a first benchmark and its performances represent the “bare minimum” for any nonlinear filtering method.

The idea of EKF is to linearize the system equations (3.1)-(3.2) around the estimated trajectory and apply the linear Algorithm 3.1 to the linearized model.

The linearized state equation has the form

\[
x(k+1) \approx f(\hat{x}(k|k), u(k), 0, k) + A(k)(x(k) - \hat{x}(k|k)) + F(k)w(k) \\
= A(k)x(k) + f(\hat{x}(k|k), u(k), 0, k) - A(k)\hat{x}(k|k) + F(k)w(k) \\
= A(k)x(k) + u^f(k) + F(k)w(k)
\]

where the Jacobian matrices

\[
A(k) := \left( \frac{\partial}{\partial x} f(x, u(k), 0, k) \right) \bigg|_{x=\hat{x}(k|k)} , \quad F(k) = \left( \frac{\partial}{\partial w} f(\hat{x}(k|k), u(k), w, k) \right) \bigg|_{w=0},
\]

and the artificial input \(u^f(k)\) are known at the prediction time, since the state estimate \(\hat{x}(k|k)\) is available (see Algorithm 3.1).

The linearized measurement equation is

\[
y(k) \approx h(\hat{x}(k|k-1), u(k), 0, k) + C(k)(x(k) - \hat{x}(k|k-1)) + G(k)v(k) \\
= C(k)x(k) + f(\hat{x}(k|k-1), u(k), 0, k) - C(k)\hat{x}(k|k-1) + G(k)v(k) \\
= C(k)x(k) + u^h(k) + G(k)v(k),
\]

where the Jacobian matrices

\[
C(k) = \left( \frac{\partial}{\partial x} h(x, u(k), 0, k) \right) \bigg|_{x=\hat{x}(k|k-1)} , \quad G(k) = \left( \frac{\partial}{\partial v} h(\hat{x}(k|k-1), u(k), v, k) \right) \bigg|_{v=0},
\]

and the artificial input \(u^h(k)\) are known at the correction time, since the state prediction \(\hat{x}(k|k-1)\) is available (see Algorithm 3.1).

Finally, the EKF reported in Algorithm 3.2 can be readily obtained by applying Algorithm 3.1 to the linearized system (3.11)-(3.12).

The computational complexity of EKF is \(O(n^3)\), without considering the Jacobian computation, which may be significant is some applications.

Because of linearization, the EKF is a suboptimal algorithm, but in practice it has been proved to work well for many applications. However, the use of EKF has two main well-known drawbacks: linearization can produce highly unstable filters if the assumptions of local linearity is violated; the derivations of the Jacobian matrices are nontrivial in most applications and often lead to significant implementation difficulties [66].

Research has attempted to adopt modifications to the basic algorithm in order to improve the performances and avoid critical behaviours. As a consequence, there is no such thing as “the” EKF, but rather there are hundreds of varieties of EKFs. In particular, 1) different coordinate systems [89, 107], 2) different factorizations of the covariance matrix, 3) second order (or
3.3. Extended Kalman Filter

1. **given**: \( f, h, \Psi_w(k), \Psi_v(k), u(k), y(k) \) \( \forall k = 0, 1, \ldots, T \)

2. **Initial estimate**: \( \hat{x}(0|−1) = \bar{x}_0, \ P(0|−1) = \Psi_{x_0} \)

3. **for** \( k = 0 \rightarrow T \ **do**

4. ▷ **Correction**

5. \( C(k) = \left( \frac{\partial}{\partial x} g(x, u(k), 0, k) \right)_{x=\hat{x}(k|k−1)} \), \( G(k) = \left( \frac{\partial}{\partial v} g(\hat{x}(k|k−1), u(k), v, k) \right)_{v=0} \)

6. \( K(k) = P(k|k−1)C^T(k) \left( C(k)P(k|k−1)C^T(k) + G(k)\Psi_v(k)G^T(k) \right)^{-1} \)

7. \( \hat{x}(k|k) = \hat{x}(k|k−1) + K(k) (y(k) - h(\hat{x}(k|k−1), u(k), 0, k)) \)

8. \( P(k|k) = (I_n - K(k)C(k)) P(k|k−1) \)

9. ▷ **Prediction**

10. \( \hat{x}(k+1|k) = f(\hat{x}(k|k), u(k), 0, k) \)

11. \( A(k) = \left( \frac{\partial}{\partial x} f(x, u(k), 0, k) \right)_{x=\hat{x}(k|k)} \), \( F(k) = \left( \frac{\partial}{\partial w} f(\hat{x}(k|k), u(k), w, k) \right)_{w=0} \)

12. \( P(k+1|k) = A(k)P(k|k)A^T(k) + F(k)\Psi_w(k)F^T(k) \)

13. **end for**

**Algorithm 3.2**: Extended Kalman Filter

Higher order) Taylor series corrections to the state vector prediction and/or the measurement update [18, 19, 49], 4) iteration of the state vector update using measurements [8], 5) different orders with which to use sequential scalar valued measurements to update the state vector, 6) tuning process noise [86], 7) quasi-decoupling, and 8) combinations of all of the above [33]. Sometimes these practical tricks result in significant improvements, but often they result in no improvement or they make performance worse.

3.3.1 EKF for Planar Tracking

The EKF can be used for planar tracking by applying Algorithm 3.2 to model (3.4)-(3.3). This is carried out through the computation of the following Jacobian matrices:

\[
A(k) = A, \quad F(k) = I_6, \quad G(k) = I_2, \quad C(k) = \frac{1}{\sqrt{x_1^2 + x_4^2}} \left[ \begin{array}{ccc} x_1 & 0 & 0 \\ -x_4 & x_1 & 0 \\ 0 & 0 & x_4 \\ 0 & 0 & -x_4 \end{array} \right]_{x=\hat{x}(k|k−1)},
\]

where \( I_n \) indicates the identity matrix in \( \mathbb{R}^n \).
The behaviour of EKF in target tracking has been widely analyzed throughout 40 years of experience. There is a general consensus that EKF suffers from significant drawbacks that make it difficult to implement and tune. In particular the EKF may easily diverge. This is firstly caused by the errors introduced by the linearization of the measurement map. However, in the tracking framework, divergence may be also caused by state model errors, despite the assumed linearity. As described in Chapter 2, modelling the tracking movement is indeed a hard issue and errors are usually not negligible. Because of these problems, tuning is often required. In the case of EKF, it consists in increasing the amount of system state noise \[\eta_w > 1\], which corresponds to use a suitable scaled covariance matrix \(\tilde{\Psi}_w = \eta_w \Psi_w\), where \(\eta_w > 1\) is a tuning parameter. The same can be applied to the measurement noise, in order to modelling the linearization error (see also comments to Example 3.1), i.e. use \(\tilde{\Psi}_v = \eta_v \Psi_v\), instead of \(\Psi_v\), with \(\eta_v > 1\). Obviously, tuning is an heuristic method which is difficult to be formally argued.

### 3.4 Unscented Kalman Filter

The UKF [65–69] has represented an interesting novelty in the nonlinear filtering field thanks to the idea of approximating the state conditional probability density function (PDF), instead of the system equations, with a bounded number of parameters. As mentioned, by Bayesian approach, the optimal solution requires that a complete description of the state conditional PDF (3.5) is maintained. Unfortunately, this exact description requires a potentially unbounded number of parameters and a number of suboptimal approximation have been proposed [31, 62, 74, 82, 104]. The UKF attempts to solve the same problem but using a bounded number of parameters.

The fundamental component of this filter is the unscented transformation (UT) which uses a set of appropriately chosen weighted points to parametrize the means and the covariance of probability distributions. For many applications, UKF has been proved to be superior to EKF and, in fact, directly comparable with a second order Gaussian filter. Further, the nature of the transform is such that the process and the observation models can be treated as “black blocks”. It is not necessary to calculate Jacobians and so the algorithm has superior implementations properties to EKF [65] and roughly the same computational complexity. As a consequence, UKF is an important benchmark to be considered in developing a new nonlinear filtering algorithm.

#### 3.4.1 The Unscented Transformation [66]

Suppose that \(x\) is a random vector with mean \(\overline{x}\) and covariance matrix \(\Psi_x\). A second random vector \(y\) is related to \(x\) through the nonlinear function

\[
y = f(x).
\]

(3.13)

Assuming the knowledge of the distribution of \(x\), the exact computation of the conditional density \(p(y|x)\) is possible with a finite number of parameters only for the linear Gaussian case. The UT is an approximate method for computing the mean \(\overline{y}\) and the covariance matrix \(\Psi_y\) of \(p(y|x)\). The objective is to design a method which yields a consistent, efficient and unbiased estimate.
The transformed statistics are consistent if the inequality

\[ \Psi_y - E \left[ (y - \bar{y}) (y - \bar{y})^T \right] \geq 0 \]  

(3.14)

holds. This condition is extremely important for the validity of the transformation method. If the statistics are not consistent, the value of \( \Psi_y \) is under-estimated. If a KF uses the inconsistent statistics, it will place too much weight on the information and under estimate the covariance, rising the possibility that the filter will diverge. By ensuring that the transformation is consistent, the filter is guaranteed to be consistent as well. However, consistency does not necessary imply usefulness because the calculated value of \( \Psi_y \) might be greatly in excess of the actual mean squared error. It is desirable that the transformation is efficient, i.e. of the left hand side of (3.14) should be minimized. Finally, it is desirable that the estimate is unbiased or \( \bar{y} \approx E[y] \).

The problem of developing a consistent, efficient and unbiased transformation procedure can be examined by considering the Taylor series expansion of (3.13) about \( \bar{x} \). This series can be expressed (using rather informal notation) as

\[ f(x) = f(\bar{x} + \delta x) \]

\[ = f(\bar{x}) + \nabla f \delta x + \frac{1}{2} \nabla^2 f \delta x^2 + \frac{1}{3!} \nabla^3 f \delta x^3 + \frac{1}{4!} \nabla^4 f \delta x^4 + \cdots \]  

(3.15)

where \( \delta x \) is a zero mean Gaussian variable with covariance matrix \( \Psi_x \), and \( \nabla^n f \delta x^n \) is the appropriate \( n \)-th order term in the multidimensional Taylor series. Taking expectations, it can be shown that the transformed mean and covariance matrix are

\[ \bar{y} = f(\bar{x}) + \frac{1}{2} \nabla^2 f \Psi_x + \frac{1}{4} \nabla^4 f E [\delta x^4] + \cdots, \]  

(3.16)

\[ \Psi_x = (\nabla f) \Psi_x (\nabla f)^T + \frac{1}{2} \cdot 4! \nabla^2 f (E[\delta x^4] - \Psi_x^2) (\nabla^2 f)^T + \]  

\[ + \frac{1}{3!} (\nabla^3 f) E [\delta x^4] (\nabla f)^T + \cdots. \]  

(3.17)

In other words, the \( n \)-th order term in the series for \( \bar{y} \) is a function of the \( n \)-th order moments of \( x \) multiplied by the \( n \)-th order derivatives of \( f(\cdot) \) evaluated at \( x = \bar{x} \). If the moments and derivatives can be evaluated correctly up to the \( n \)-th order, the mean is correct up to the \( n \)-th order as well. Similar comments hold for the covariance equation as well, although the structure of each term is more complicated. Since each term in the series is scaled by a progressively smaller and smaller term, the lowest order terms in the series are likely to have the greatest impact. Therefore, the prediction procedure should be concentrated on evaluating the lower order terms.

Linearization assumes that the second and higher order terms of \( x \) in Equation (3.15) can be neglected. Under this assumption,

\[ \bar{y} \approx f(\bar{x}) \]

\[ \Psi_y \approx (\nabla f) \Psi_x (\nabla f)^T. \]
Comparing these expressions with (3.16) and (3.17), it is clear that these approximations are accurate only if the second and higher order terms in the mean and fourth and higher order terms in the covariance are negligible. However, in many practical situations linearization introduces significant biases or errors. An extremely common and important problem is the transformation between polar and Cartesian coordinate systems [64, 77], which actually is the tracking problem transformation. An example of this transformation follows.

Example 3.1

Let the polar coordinates of a point \( x \in \mathbb{R}^2 \) be Gaussian distributed with mean \([1m, \pi/2\text{rad}]^T\) and covariance matrix \( \Psi_x = \text{diag}([0.02m, \pi/12\text{rad}]) \). The transformation between polar to Cartesian coordinates is defined as follows:

\[
\begin{align*}
y_1 &= x_1 \cos(x_2), \\
y_2 &= x_1 \sin(x_2).
\end{align*}
\] (3.18)

Figure 3.1 shows the mean and the standard deviation ellipses for the actual and linearized form of the transformation. The standard deviation ellipse is defined as the locus of points \( \{ y : (y - \bar{y}) \Psi_y^{-1} (y - \bar{y})^T = 1 \} \) and is a representation of the size and orientation of \( \Psi_y \).

As shown in the example, the linearized transformation is biased and inconsistent. This is most pronounced in the range direction. Linearization errors effectively introduce an error.

---

3This example has been completely taken from [66].
which is over 1.5 times the standard deviation of the range measurement. Since a bias arises from the transformation process itself, the same error with the same sign will be committed each time a coordinate transformation takes place. Even if there were no bias, the transformation is inconsistent. Its ellipse is not long enough in the range direction. In fact, the nature of the inconsistency compounds the problem of the biased-ness: not only the estimate or range is in error, but also its estimated mean squared error is much smaller than the true value.

In practice, the inconsistency can be resolved by introducing additional stabilising noise which increases the size of the transformed covariance. This is one possible of why EKFs are so difficult to tune: sufficient noise must be introduced to offset the defects of linearization. However, introducing stabilising noise is an undesirable solution since the estimate remains biased and there is no general guarantee that the transformed estimate remains consistent or efficient. A more accurate prediction algorithm is required.

The UT attempts to avoid the drawbacks of linearization with the idea of approximating the conditional density \( p(x|y) \) instead of the nonlinear function \( f(\cdot) \). The approach is illustrated in Figure 3.2. A set of points (or sigma points) are chosen so that their sample mean and sample covariance are \( \bar{x} \) and \( \Psi_x \). The nonlinear function is applied to each point in turn to yield a cloud of transformed points and \( \bar{y} \) and \( \Psi_y \) are the statistics of the transformed points. Although this method bares a superficial resemblance to Monte Carlo-type methods (see Section 3.5), there is an extremely important and fundamental difference. The samples are not drawn at random but rather according to a specific, deterministic algorithm. Since the problems of statistical convergence are not an issue, high order information about the distribution can be captured using only a very small number of points.

The \( n_x \)-dimensional random variable \( x \) with mean \( \bar{x} \) and covariance \( \Psi_x \) is approximated by \( 2n_x + 1 \) weighted points given by

\[
\begin{align*}
x^{(0)} &= \bar{x}, \\
x^{(i)} &= \bar{x} + \sqrt{\frac{n_x}{1-\omega^{(i)}}} u_i, \\
x^{(i+n_x)} &= \bar{x} - \sqrt{\frac{n_x}{1-\omega^{(i)}}} u_i, \\
\omega^{(0)} &= 1 - \omega^{(i)}, \\
\omega^{(i+n_x)} &= 1 - \omega^{(i)},
\end{align*}
\]

(3.19)

where \( i = 1, 2, \ldots, n_x \), \( u_i \) is the \( i \)-th row or column of the matrix square root of \( \Psi_x \) and \( \omega^{(i)} \) is the weight associated with the \( i \)-th point. The weight \( \omega^{(0)} \) is a tuning parameter.

![Figure 3.2: The principle of unscented transform.](image)
Once the sigma points are chosen, the approximations of $\bar{y}$ and $\Psi_y$ are computed as weighted means. Denote the transformed sigma points with

$$y^{(i)} = f(x^{(i)})$$

for $i = 1, 2, \ldots, n_x$ and associate them with the weights $\omega^{(i)}$. The estimated mean follows as

$$\bar{y} \approx \sum_{i=0}^{2n_x} \omega^{(i)} y^{(i)}$$

and the covariance matrix as

$$\Psi_y \approx \sum_{i=0}^{2n_x} \omega^{(i)} (y^{(i)} - \bar{y})(y^{(i)} - \bar{y})^T.$$  (3.20)

The detailed properties of this algorithm have been studied in [65–67]. A summary of the results follows:

1. Since the mean and covariance of $x$ are captured precisely up to the second order, the calculated values of the mean and covariance of $y$ are correct to the second order as well. This means that the mean is calculated to a higher order of accuracy than the linearization, whereas the covariance is calculated to the same order of accuracy. However, there are further performance benefits. Since the distribution of $x$ is being approximated rather than $f(\cdot)$, its series expansion is not truncated at a particular order. It can be shown that the unscented algorithm is able to partially incorporate information from the higher orders, leading to even greater accuracy. However, very little is said about how and under what conditions this holds [56].

2. The sigma points capture the same mean and covariance irrespective of the choice of matrix square root which is used. Numerically efficient and stable methods such as the Cholesky decomposition can be used [66].

3. The mean and covariance matrix are calculated using standard vector and matrix operations. This means that the algorithm is suitable for any choice of process model, and implementation is extremely rapid because it is not necessary to evaluate the Jacobians which are needed in an EKF.

4. A small value of $\omega^{(0)}$ moves the sigma points away from the mean, whereas a large value gathers them closer to the center. This allows for tuning of the UT. The weight $\omega^{(0)} = 1 - n_x/3$ gives, according to [66, 68], preferable properties for Gaussian distributions. It is also possible to use other sets of sigma points and/or parameterizations to change the behaviour of the approximation and this way get more degrees of freedom for tuning [56, 68].

5. Sometimes the UT produces indefinite covariance matrix estimates. As a way to solve this, and to also improve the performance further, techniques have been proposed to slightly change the weights and the placement of the sigma points. The most common
variation, combining two modifications [68, 110], introduces an extra weight on the mean sigma point when computing the covariance and a different set of tuning parameters. The covariance estimate (3.20) then becomes

\[
\Psi_y \approx 2n_x \sum_{i=0}^{2n_x} \omega^{(i)} (y^{(i)} - \bar{y}) (y^{(i)} - \bar{y})^T + (1 - \alpha^2 + \beta) (y^{(0)} - \bar{y}) (y^{(0)} - \bar{y})^T
\]  

(3.21)

with \( \omega^{(0)} \) chosen as \( \omega^{(0)} = \lambda / (n_x + \lambda) \) where \( \lambda = \alpha^2 (n_x + \kappa) - n_x \).

It is common to introduce the concept of different weights for mean and covariance approximation. Hence, use \( \omega_m^{(i)} \) for the mean and \( \omega_c^{(i)} \) for the covariance, and the only difference between the two sets of weights being \( \omega_c^{(0)} = \omega_m^{(0)} + (1 - \alpha^2 + \beta) \). There are three new parameters introduced with this modification:

- \( \alpha \) - is a primary scaling factor determining how far from the center the sigma points should be placed. The recommendation is \( \alpha \approx 10^{-3} \), i.e., keep the sigma points close to the center;
- \( \beta \) - is a distribution compensation taking care of higher order effects. In [68], \( \beta = 2 \) is claimed to be optimal for Gaussian distributions;
- \( \kappa \) - is a secondary scaling factor, usually chosen to \( \kappa = 0 \).

Some observations; the weights for the covariance estimate sums up to more than one, and \( \omega_m^{(0)} = 1 - 1/\alpha^2 \ll 0 \) if \( \kappa = 0 \) and \( \alpha \ll 1 \), which differs substantially from the suggestion without the covariance matrix.

The performance benefits of using UT can be seen in Figure 3.3 which shows the mean and the standard deviation ellipses about the transformation from polar to Cartesian coordinates (3.18), introduced by Example 3.1. It can be noted that the unscented mean value is the same as the true value, i.e., on the scale of the graph, the two points lie on top of the one other. Further, the UT is consistent. In fact, its ellipse is slightly larger than the true one.

3.4.2 The Filtering Algorithm

The UT is now used to select points from an augmented state space that includes the system state \( x(k) \), the the process noise \( w(k) \), and the measurement noise \( v(k) \) in one augmented state vector,

\[
\mathcal{X}(k) = \begin{bmatrix} x(k) \\ w(k) \\ v(k) \end{bmatrix}
\]  

(3.22)

with the dimension \( n_x \). Using the augmented state vector \( \mathcal{X} \) allows for a straightforward utilization of the UT especially when it comes to finding suitable sigma points. However, once the sigma points have been acquired the augmented state can be split up again to give a more familiar notation. By letting the sigma points pass through the model dynamics the following prediction phase is obtained:

\[
x^{(i)}(k + 1|k) = f \left( x^{(i)}(k|k), u(k), w^{(i)}(k), k \right),
\]
and the result can be combined to obtain the estimate

\[ \hat{x}(k + 1|k) = \sum_{i=0}^{2n_x} \omega^{(i)}(k) x^{(i)}(k + 1|k) \]

\[ P(k + 1|k) = \sum_{i=0}^{2n_x} \omega^{(i)}(k) \left( x^{(i)}(k + 1|k) - \hat{x}(k + 1|k) \right) \left( x^{(i)}(k + 1|k) - \hat{x}(k + 1|k) \right)^T \]

The information in the measurements is introduced in a similar way by first obtaining the predicted measurements based on the sigma points

\[ y^{(i)}(k) = h \left( x^{(i)}(k|k - 1), u(k), v^{(i)}(k), k \right), \]

yielding

\[ \hat{y}(k) = \sum_{i=0}^{2n_x} \omega^{(i)}(k) y^{(i)}(k) \]

\[ P_{yy}(k|k - 1) = \sum_{i=0}^{2n_x} \omega^{(i)}(k) \left( y^{(i)}(k) - \hat{y}(k) \right) \left( y^{(i)}(k) - \hat{y}(k) \right)^T \]
The filter gain is computed as the cross-covariance matrix between state and measurement:

\[ P_{xy}(k|k-1) = \sum_{i=0}^{2n_X} \omega^{(i)}(k) \left( x^{(i)}(k|k-1) - \hat{x}(k|k-1) \right) \left( y^{(i)}(k) - \hat{y}(k) \right)^T, \]

multiplied with the inverse state covariance matrix, as this will project the state onto the new innovation:

\[ K(k) = P_{xy}(k|k-1) P_{yy}^{-1}(k|k-1) \]

and the estimate follows as

\[ \hat{x}(k|k) = \hat{x}(k|k-1) + K(k) (y(k) - \hat{y}(k)) \]
\[ P(k|k) = P(k|k-1) - P_{xy}(k|k-1) P_{yy}^{-1}(k|k-1) P_{xy}^T(k|k-1). \]

The UKF is given in detail in Algorithm 3.3. The computational complexity of UKF is \( O(n^3) \).

It is of course possible to use the modified form of the unscented transform with different weights for mean and covariance computations [41]. A question not dealt with in detail here is how often new sigma points should be generated. In Algorithm 3.3 it is done once every time through the filtering loop (line 13), but it is also possible to do it before both measurement update and time update. However, if the transformation in the time update results in an asymmetric distribution, the sigma points will represent this fact, while the Gaussian approximation does not. Hence, generating new sigma points from the Gaussian approximation before the measurement update will lose information compared to keeping the ones from the time update. Note that the same is not true for the measurement update since new sigma points must be generated to represent the distributions after the filtering.

Furthermore, if the noise enters the system linearly it is possible to use this structure to improve performance since then the contributions from the noise can be handled analytically at low cost. More about this can be found in [110, 113].

### 3.4.3 UKF for Planar Tracking

The UKF can be used for planar tracking by applying Algorithm 3.3 to model (3.4)-(3.3), where the sigma point setting (line 13) is computed using Gaussian hypothesis. The noises sequences are indeed usually supposed to be Gaussian or, anyway, symmetrically distributed. This means that the augmented state \( \mathcal{X}(k) \), defined in (3.22), is supposed to have the following mean value and covariance matrix:

\[ \bar{\mathcal{X}}(k) := E[\mathcal{X}(k)] = \begin{bmatrix} \hat{x}(k|k) \\ 0_{6 \times 6} \\ 0_{2 \times 6} \end{bmatrix}, \]
\[ \Psi_{\mathcal{X}}(k) := E \left[ (\mathcal{X}(k) - \bar{\mathcal{X}}(k)) (\mathcal{X}(k) - \bar{\mathcal{X}}(k))^T \right] = \begin{bmatrix} P(k|k) & 0_{6 \times 6} & 0_{6 \times 2} \\ 0_{6 \times 6} & F \Psi_{w} F^T & 0_{6 \times 2} \\ 0_{2 \times 6} & 0_{2 \times 6} & \Psi_{v} \end{bmatrix}, \]
3.4. Unscented Kalman Filter

1. **given**: $f, h, \Psi_{w}(k), \Psi_{v}(k), u(k), y(k) \ \forall k = 0, 1, \ldots, T$

2. Initial estimate: $\hat{x}(0|1) = \bar{x}_{0}, \ P(0|1) = \Psi_{x_{0}}$

3. Choose sigma points as described in line 13

4. **for** $k = 0 \rightarrow T$ **do**

5. ▷ **Correction**

6. $y^{(i)}(k) = h \left(x^{(i)}(k|k-1), u(k), v^{(i)}(k), k\right)$

7. $\hat{y}(k) = \sum_{i=0}^{2n_{x}} \omega^{(i)}(k)y^{(i)}(k)$

8. $P_{yy}(k|k-1) = \sum_{i=0}^{2n_{x}} \omega^{(i)}(k) \left(y^{(i)}(k) - \hat{y}(k)\right) \left(y^{(i)}(k) - \hat{y}(k)\right)^{T}$

9. $P_{xy}(k|k-1) = \sum_{i=0}^{2n_{x}} \omega^{(i)}(k) \left(x^{(i)}(k|k-1) - \hat{x}(k|k-1)\right) \left(y^{(i)}(k) - \hat{y}(k)\right)^{T}$

10. $K(k) = P_{xy}(k|k-1)P_{yy}^{-1}(k|k-1)$

11. $\hat{x}(k|k) = \hat{x}(k|k-1) + K(k) \left(y(k) - \hat{y}(k)\right)$

12. $P(k|k) = P(k|k-1) - P_{xy}(k|k-1)P_{yy}^{-1}(k|k-1)P_{xy}^{T}(k|k-1)$

13. Choose sigma points, $X^{(i)}(k)$ partitioned as in (3.22), and weights $\omega^{(i)}(k)$ by (3.19)

14. ▷ **Prediction**

15. $x^{(i)}(k+1|k) = f \left(x^{(i)}(k|k), u(k), w^{(i)}(k), k\right)$

16. $\hat{x}(k+1|k) = \sum_{i=0}^{2n_{x}} \omega^{(i)}(k)x^{(i)}(k+1|k)$

17. $P(k+1|k) = \sum_{i=0}^{2n_{x}} \omega^{(i)}(k) \left(x^{(i)}(k+1|k) - \hat{x}(k+1|k)\right) \left(x^{(i)}(k+1|k) - \hat{x}(k+1|k)\right)^{T}$

18. **end for**

Algorithm 3.3: Unscented Kalman Filter

where $0_{n}$ and $0_{n \times m}$ indicate the null vector in $\mathbb{R}^{n}$ and the null matrix in $\mathbb{R}^{n \times m}$, respectively. Sigma points $X^{(i)}(k)$ can be hence generated by applying rule (3.19) to the random vector $X(k)$ with the above reported mean value and covariance matrix.

As for EKF, tuning may be required to use UKF for planar tracking. In this case more degrees of freedom are available. The same tuning of EKF with parameters $\eta_{w}$ and $\eta_{v}$ can be indeed used (see Section 3.3.1). Moreover, literature suggests to use the modified UKF, where parameters $\alpha$, $\beta$, and $\kappa$, are introduced (see (3.21)). These parameters open to the possibility
of a fine tuning. However, under Gaussian hypothesis the recommended values are $\alpha = 10^{-3}$, $\beta = 2$, and $\kappa = 0$, as previously mentioned.

3.5 Particle Filter

As for UKF, the idea of PFs [11, 12, 23, 36, 38, 46, 50, 51, 53, 58, 59, 61, 63, 72, 80, 87, 90, 95] is to approximate the state conditional PDF. The approach consists in using Monte Carlo (MC) methods [22, 37, 39, 54, 56, 73, 76, 84, 85] for approximating the statistics of transformed random variable, which roughly consists in draw samples from the original distribution, apply the function, and see what happens. The resulting samples should, in some way represent the new distribution. As mentioned in Chapter 1, the estimate accuracy grows with the number of samples [39, 96]. More formally, as the samples number tends to infinity the estimate becomes closer to the optimal one. However, with a relative large number of samples, the complexity could become enormous and make the solution not implementable [34]. Because of these features, PFs should be considered as fundamental benchmark in the analysis of nonlinear filtering methods. In this case, the computational time has to be also evaluated, in order to obtain a consistent comparison.

Similarly to what done for UKF, the MC approach is firstly described as general method for approximating probability densities. Then, the filtering procedure is introduced.

3.5.1 The Monte Carlo Transformation [56]

Consider the same problem of Section 3.4.1 where the UT has been used to approximate the mean value and covariance matrix of a random vector $y$, related to an other $n_x$-dimensional random vector $x$ (with mean $\bar{x}$ and covariance matrix $\Psi_x$) by a nonlinear function $y = f(x)$.

The distribution of $x$ is approximated using a set of $N_p \gg 1$ of independent and identically distributed (IID) samples $\{x^{(i)}, i = 1, 2, \ldots, N_p\}$, also called particles:

$$p(x) \approx \sum_{i=0}^{N_p} \omega^{(i)} \delta \left( x - x^{(i)} \right),$$

where, for now, $\omega^{(i)} = \frac{1}{N_p}$ are the particle weights, and $\delta(\cdot)$ is the Dirac delta function.

Particles for the target distribution $p(y)$ can be obtained applying $f(\cdot)$ to all particles about $x$, i.e.

$$y^{(i)} = f(x^{(i)}),$$

and form the target distribution as

$$p(y) \approx \sum_{i=0}^{N_p} \omega^{(i)} \delta \left( y - y^{(i)} \right).$$

The approximation is the same as for $p(x)$, and the density of particles in a region gives the actual PDF. The mean and the covariance of $p(y)$ can be obtained as the sample mean and

---

$^4$Recall that for a fixed $x$, $p(y|x) = p(y)$. 

covariance matrix:

\[
\bar{y} \approx \sum_{i=0}^{N_p} \omega^{(i)} y^{(i)},
\]

\[
\Psi_y \approx \sum_{i=0}^{N_p} \omega^{(i)} \left( y^{(i)} - \bar{y} \right) \left( y^{(i)} - \bar{y} \right)^T.
\]

It is clear that the larger the set of particle used, the better the approximation becomes.

The theory behind MC integration [96] can be used to validate this method. Such a technique uses statistical properties to compute integrals that otherwise are difficult to handle. The idea is to reformulate difficult integrals on a form where computing an expected value renders the integral of interest. To illustrate this, consider the integral

\[
I = \int f(x)dx = \int g(x)p(x)dx,
\]

where \( p \) should be a proper PDF and \( g(x) = f(x)/p(x) \). The value of the integral \( I \) can then be approximated with the sum

\[
\hat{I}_{N_p} := \sum_{i=1}^{N_p} \frac{1}{N_p} g(x^{(i)}),
\]

where \( \{x^{(i)}, i = 1, 2, \ldots, N_p\} \) are \( N_p \) IID samples form the distribution given by \( p \). The approximation uses that \( I = E_p [g(x)] \), where \( E_p \) indicates the mean value under the distribution \( p \), and that an expected value can be approximated with a sample mean. Furthermore, it follows from the low of large numbers that if the variance over \( p \), \( \text{var}_p [g(x)] = \Psi < +\infty \), then

\[
\lim_{N_p \to +\infty} \sqrt{N_p} \left( \hat{I}_{N_p} - I \right) \sim N(0, \Psi),
\]

i.e. \( \hat{I}_{N_p} \to I \) as \( N \to +\infty \) and the quality of the estimate improves with increasing \( N_p \), [39, 96]. Note, the convergence is in theory independent of the dimension of \( x \), and MC integration should hence suffer little from the curse of dimensionality in contrast to deterministic approximative integration methods. However, this is according to [34] overly optimistic and MC integration is claimed to suffer from the curse of dimensionality. This, on the other hand, seems too pessimistic for most applications in practice.

It may be difficult, or even impossible, to draw samples from \( p \). This is sometimes the case with the posterior state distributions used later. If this is the problem, choose another proper PDF \( q \) such that \( p(x) > 0 \) implies \( q(x) > 0 \) for \( x \) in the domain of \( p \), i.e., the support of \( p \) is included in the support of \( q \). Using \( q \) for generating particles yields

\[
\hat{I}_{N_p} = \sum_{i=0}^{N_p} \frac{p(x^{(i)})}{q(x^{(i)})} g(x^{(i)}) = \sum_{i=0}^{N_p} \omega^{(i)} g(x^{(i)}),
\]

with the same limit and principal convergence as before. The distribution given by \( q \) is often
Figure 3.4: The mean and standard deviation ellipses for the actual and MC approximation forms of the transformation from polar to Cartesian coordinates (Example 3.1). The true mean is at × and the uncertainty ellipse is solid. The positions of the mean values of the MC approximations with $N_p = 3.5 \times 10^2$, $3.5 \times 10^3$, and $3.5 \times 10^4$ are indicated by ○, □, and +, respectively. The ellipses are dotted, point-dashed, and dashed, respectively.

called proposal distribution and $\omega^{(i)}$ importance weights. Note that even if $p$, and thus $\omega^{(i)}$, is only known up to a normalizing constant, this is not a problem since

$$\lim_{N_p \to +\infty} \sum_{i=0}^{N_p} \omega^{(i)} = \int c \frac{p(x)}{q(x)} dx = \int cp(x) dx = c,$$

and it is hence possible to normalize the distribution and compute the integral with

$$\hat{I}_{N_p} = \frac{\sum_{i=0}^{N_p} \omega^{(i)} g(x^{(i)})}{\sum_{i=0}^{N_p} \omega^{(i)}}.$$

Figure 3.4 shows the approximations obtained by using the MC transformation for representing a distribution after the change from polar to Cartesian coordinates (3.18), introduced in Example 3.1. In particular, this figure reports the mean values and standard deviation ellipses (see Example 3.1) of the actual transformed distribution, and those computed using MC approximations with an increasing number of particles. Notice that as this number increases, the estimate becomes more consistent and bias is reduced. More precisely, $N_p = 350$ is not enough to obtain a consistent and unbiased estimate, since the true ellipse is not completely enclosed in the estimated one and the mean is significantly far from the true location. Moreover, notice that a minimum of 35000 particles is required to obtain a result similar to that of the UT, shown in Figure 3.3.
3.5.2 The Filtering Algorithm

PF is a suboptimal solution to the Bayesian formulation of nonlinear filtering, which requires to recursively calculate the conditional PDF \( p(x(k)|Y_k) \) at time \( k \), as briefly illustrated in the next.

It is assumed that the initial PDF \( p(x(0)|Y_{-1}) = p(x_0) \) (where \( Y_{-1} \) indicates the void measurement sequence) of the state vector, which is also known as prior, is available (according to Assumption (3.b)). Then, in principle, \( p(x(k)|Y_k) \) may be obtained, recursively, by a correction and a prediction steps, exactly as for KF, EKF and UKF.

Suppose the prior PDF \( p(x(k)|Y_{k-1}) \) to be available at time \( k \), together with the measurement \( y(k) \). The latter can be used to update the current state PDF via Bayes’ rule:

\[
p(x(k)|Y_k) = \frac{p(y(k)|x(k))p(x(k)|Y_{k-1})}{p(y(k)|Y_{k-1})},
\]

(3.23)

where the normalizing term

\[
p(y(k)|Y_{k-1}) = \int p(z(k)|x(k)) p(x(k)|Y_{k-1}) \, d(x(k))
\]

depends on the likelihood function \( p(y(k)|x(k)) \) defined by the measurement model (3.2) and the known statistics of \( \{v(k)\} \) (see Assumption (3.a)). Equation (3.23) constitutes the Bayesian correction stage, since the measure \( y(k) \) is used to correct the estimate \( p(x(k)|Y_k) \).

The Bayesian prediction stage involves using the system model (3.1) to obtain the prior PDF of the state at time \( k+1 \). In this case, the Chapman-Kolmogorov equation is used:

\[
p(x(k+1)|Y_k) = \int p(x(k+1)|x(k)) p(x(k)|y(k)) \, d(x(k)).
\]

(3.24)

Notice that in (3.24), use has been made of the fact that \( p(x(k+1)|x(k), Y_k) = p(x(k+1)|x(k)) \) as (3.1) describes a Markov process of order one. The probabilistic description of the state evolution \( p(x(k+1)|x(k)) \) is defined by the system equation (3.1) and the known statistics of \( \{w(k)\} \) (see Assumption (3.a)).

The PF uses the MC integration technique and the MC transformation, described in Section 3.5.1, to approximate the prior and current PDFs of the state vector \( x(k) \).

As in the above reported Bayesian formulation, suppose the prior PDF \( p(x(k)|Y_{k-1}) \) to be available at time \( k \) in the MC form\(^5\):

\[
p(x(k)|Y_{k-1}) \approx \sum_{i=0}^{N_p} \omega_{k|k-1}^{(i)} \delta \left( x(k) - x_k^{(i)} \right),
\]

where particles \( \{x_k^{(i)} \}, i = 1, 2, \ldots, N_p \) is a set of \( N_p \) IID samples from a proposal distribution, and \( \{\omega_{k|k-1}^{(i)} \}, i = 1, 2, \ldots, N_p \) are the associated importance weights.

\(^5\)For readability, in this section, time is subscripted for particles and importance weights, differently of the generally used notation.
Now, the measurement $y(k)$ must be used to update the state conditional PDF, which has the form:

$$p(x(k)|Y_k) \approx \sum_{i=0}^{N_p} \omega_{k|i,k}^{(i)} \delta \left( x(k) - x_k^{(i)} \right). \quad (3.25)$$

Notice that the update consists on the computation of the importance weights $\{\omega_{k|i,k}^{(i)} \mid i = 1, 2, \ldots, N_p \}$. According to (3.23),

$$p(x(k)|Y_k) \propto p(y(k)|x(k)) \cdot p(x(k)|Y_{k-1}),$$

so that, the equivalence of (3.25) and (3.26) yields the weights update:

$$\omega_{k|i,k}^{(i)} = \frac{p \left( y(k)|x_k^{(i)} \right) \omega_{k|i,k-1}^{(i)}}{\sum_{j=1}^{N_p} p \left( y(k)|x_k^{(j)} \right) \omega_{k|i,k-1}^{(j)}}. \quad (3.27)$$

The prediction stage consists in computing $p(x(k+1)|Y_k)$ by using (3.24). To do this, a new set of IID particles $\{x_{k+1}^{(i)} \mid i = 1, 2, \ldots, N_p \}$ is drawn by an importance distribution $q(x_{k+1}^{(i)} | x_k^{(i)}, Y_k)$. The associated importance weights are obtained by the following update:

$$\omega_{k+1|i,k}^{(i)} = \frac{p \left( x_{k+1}^{(i)}|x_k^{(i)} \right) \omega_{k|i,k}^{(i)}}{q \left( x_{k+1}^{(i)}|x_k^{(i)}, Y_k \right) \omega_{k|i,k}^{(i)}},$$

yielding the approximate prior PDF

$$p(x(k+1)|Y_k) \approx \sum_{i=0}^{N_p} \omega_{k+1|i,k}^{(i)} \delta \left( x(k+1) - x_{k+1}^{(i)} \right). \quad (3.28)$$

At any time in the algorithm, the approximated minimum variance estimate of the state can be obtained as weighted sample mean

$$\hat{x}(k|\tau) = E_{x(k)|Y_\tau} [x(k)] \approx \sum_{i=0}^{N_p} \omega_{k|\tau}^{(i)} x_k^{(i)}. \quad (3.29)$$

The corresponding covariance matrix is given by the sample covariance matrix

$$P(k|\tau) = \text{var}_{x(k)|Y_\tau} [x(k)] \approx \sum_{i=0}^{N_p} \omega_{k|\tau}^{(i)} \left( x_k^{(i)} - \hat{x}(k|\tau) \right) \left( x_k^{(i)} - \hat{x}(k|\tau) \right)^T. \quad (3.30)$$

Notice that if the proposal distribution is chosen to be $q(x_{k+1}^{(i)}|x_k^{(i)}, Y_k) = p(x_{k+1}^{(i)}|x_k^{(i)})$, this simplifies the update of the importance weights to $\omega_{k+1|i,k}^{(i)} = \omega_{k|i,k}^{(i)}$. However, this is not optimal with respect the statistical properties of the filter. Nevertheless, due to its simplicity the simple proposal is often used.
1. given: $u(k)$, $y(k)$, $p(y(k)|x(k))$, and $p(x(k+1)|x(k))$, $\forall k = 0, 1, \ldots, T$

2. Initial estimate: $\{x_0^{(i)}, i = 1, \ldots, N_p\} \sim p(x_0)$, and $\{\omega_0^{(i)}, i = 1, \ldots, N_p\} = \frac{1}{N_p}$

3. for $k = 0 \rightarrow T$ do

4. \hspace{1em} $\triangleright$ Correction

5. $\omega_k^{(i)} = \frac{p\left(y(k)|x_k^{(i)}\right) \omega_{k-1}^{(i)}}{\sum_{j=1}^{N_p} p\left(y(k)|x_j^{(j)}\right) \omega_{k-1}^{(j)}}$, $i = 1, 2, \ldots, N_p$

6. $\hat{x}(k|k) = \sum_{i=0}^{N_p} \omega_k^{(i)} x_k^{(i)}$

7. $P(k|k) = \sum_{i=0}^{N_p} \omega_k^{(i)} (x_k^{(i)} - \hat{x}(k|k))(x_k^{(i)} - \hat{x}(k|k))^T$

8. \hspace{1em} $\triangleright$ Resampling See Algorithms 3.5 and 3.6 for SIR and SIS resampling

9. \hspace{1em} $\triangleright$ Prediction

10. Generate $N_p$ particles $x_{k+1}^{(i)} \sim q(x_{k+1}^{(i)}|x_k^{(i)}, Y_k)$, $q$ being an importance distribution.

11. $\omega_{k+1|k}^{(i)} = \frac{p(x_{k+1}^{(i)}|x_k^{(i)})}{q(x_{k+1}^{(i)}|x_k^{(i)}, Y_k)} \omega_k^{(i)}$, $i = 1, 2, \ldots, N_p$

12. end for

Algorithm 3.4: Particle Filter

The computational complexity of the generic PF algorithm is $O(N_p n^2)$. If compared with that of EKF and UKF, the computational burden of PFs may be significantly heavier. For example, an usual scenario has $n \approx 10$ and $N_p \approx 10^3$, so that PFs is 100 times slower than EKF and UKF. Notice that the given complexity does not include the computation of resampling, which will be discussed in the next section.

The above is a sequential MC method and represents what was available in the 1950s [55]. It can be shown that using this method the approximated distribution will degenerate so that only a few particles actually contribute to the description of the PDF [1, 39, 56, 73]. Using $q(x_{k+1}^{(i)}|x_k^{(i)}, Y_k) = p(x_{k+1}^{(i)}|x_k^{(i)})$ this can happen quite quickly, whereas a more careful choice of importance distribution may at best slow down the process.

A solution to the problem is to introduce a resampling step as suggested in [51]. The resampling step in combination with the increased computational power was what was needed to turn the PF into an interesting method. Algorithm 3.4 represents a generic PF with resampling.
3.5.3 Resampling

The resampling step re-generates the particles used to represent the PDF in the PF so that the stochastic support is maintained. That is, given

\[ p(x) \approx \sum_i \omega^{(i)} \delta \left( x - x^{(i)} \right), \]

find and approximation

\[ p_+ \approx \sum_i \omega^{(i)}_+ \delta \left( x - x^{(i)}_+ \right) \]

with a better support. Usually, \( \omega^{(i)}_+ = \frac{1}{N_p} \) and \( x^{(i)}_+ \) are selected from the original set of particles \( x^{(i)} \). Several methods have been suggested to do this. As done in [56], four different algorithms are here considered.

Resampling is performed because the stochastic support of the particle set is lost since the particles lose importance over time. The resampling avoids this by randomly selecting new particles from the old set paying attention to how likely they are, i.e., drawing new particles from the old ones with replacement and \( P(x^{(i)} = x^{(i)}) = \omega^{(i)} \). More mathematically, let \( x^{(j)}_+ = x^{(i)} \) with

\[ i = C^{-1}(u^{(j)}) \]

where \( u^{(j)} \) is a random number \( u^{(j)} \in [0, 1) \), and \( C^{-1} \) the generalized inverse cumulative density function (CDF) for the particles, \( C(i) := \sum_{j=1}^{i} \omega^{(j)} \). Different sampling methods differ in how the \( u^{(j)} \) are generated.

- **Multinominal resampling** The random numbers \( u^{(j)} \) are selected in a completely random fashion from a uniform distribution,

\[ u^{(j)} \sim U([0, 1)). \]

This is the method used in the first papers about PF. It is possible to implement this resampling procedure in \( O(N_p) \) operations by sampling \( N_p \) ordered uniforms using an algorithm based on order statistics [1, 94].

- **Stratified resampling** The random numbers \( u^{(j)} \) are drawn as

\[ u^{(j)} = \frac{j - 1 + \tilde{u}^{(j)}}{N_p}, \quad \tilde{u}^{(j)} \sim U([0, 1)). \]

Generated in this way, \( u^{(j)} \) are always ordered and no extra measures need to be taken in order to ensure that. Furthermore, it guarantees that if the inverse CDF is split in \( N_p \) equally large parts, there will be one sample from each. This somewhat restricts the randomness as there is some structure in the \( u^{(j)} \) values, for instance there cannot be two \( u^{(j)} < 1/N_p \). This may improves the result of the resampling [56].
3.5. Particle Filter

Figure 3.5: Illustration of how \( u(j) \) are chosen with multinominal, stratified, and systematic resampling, in this case with 10 particles (Figure Courtesy of [56]).

- **Systematic resampling** The random numbers \( u(j) \) are evenly spaced according to

\[
  u(j) = \frac{j - 1 + \tilde{u}}{N_p}, \quad \tilde{u} \sim U([0, 1)).
\]

That is, as with the stratified sampling, there is one particle selected from each segment of length \( 1/N_p \).

- **Residual resampling** With this resampling, particle \( i \) should be chosen \( n^{(i)} = \lceil N_p \omega^{(i)} \rceil \)
6 times, giving \( N^* = \sum_{i=1}^{N_p} n^{(i)} \) new particles. The remaining \( N - N^* \) particles are then chosen using another resampling algorithm. Choosing the first \( N^* \) particles is very fast, and using this method could hence improve the computational complexity.

Figure 3.5, taken from [56], clearly shows the differences between the resampling strategies in determining the new set of particles. Resampling complexity is studied in [16, 17] from a more hardware near point of view.

The difference between the Sampling Importance Resampling (SIR) and Sequential Importance Sampling (SIS) PF is how often resampling is conducted, the resampling itself can be done using any resampling algorithm but the original formulations use multinominal resampling. When using SIR resampling (Algorithm 3.5) the particles are resampled every time the estimation loop is performed, whereas in the generalization, SIS (Algorithm 3.6), the particles are only resampled when it is necessary according to some measure. An often used measure of the particle quality is the effective sample size \([1, 11, 53, 73]\):

\[
  N_{eff} := \frac{N_p}{1 + \text{var} (\omega^{(i)})},
\]

but other measures have been suggested. The effective sample size indicates how many of the particles actually contribute to the support of the studied PDF. If \( N_{eff} < N_p \) this indicates that the support is poor and that resampling is needed to avoid degeneration of the filter. Unfortunately, \( N_{eff} \) is difficult to calculate analytically, but can be approximated with

\[
  \hat{N}_{eff} := \frac{1}{\sum_i (\omega^{(i)})^2}.
\]

\(^6\lfloor \cdot \rfloor \) indicates the integer part or floor function.
1. Use e.g., multinomial, stratified, systematic, or residual resampling to get the new set of particles \( \{ x_k^{(i)}, i = 1, 2, \ldots, N_p \} \)

2. \( \omega_k^{(i)} = \frac{1}{N_p}, \ \forall \ i = 1, 2, \ldots, N_p \)

Algorithm 3.5: Sampling Importance Resampling (SIR)

1. **Given:** \( N_{th} \)
2. \( \hat{N}_{eff} = \frac{1}{\sum_i (\omega_k^{(i)})^2} \)
3. **if** \( \hat{N}_{eff} < N_{th} \) **then**
4. get a new set of particles \( \{ x_k^{(i)}, i = 1, 2, \ldots, N_p \} \) by some resampling method
5. \( \omega_k^{(i)} = \frac{1}{N_p}, \ \forall \ i = 1, 2, \ldots, N_p \)
6. **end if**

Algorithm 3.6: Sequential Importance Sampling (SIS)

Therefore, \( \hat{N}_{eff} \) can be used to determine when to resample, i.e. given some threshold value \( N_{th} \), resampling is required when \( N_{eff} \approx \hat{N}_{eff} < N_{th} \). A common choice of \( N_{th} \) is \( 2/3N_p \), also for tracking applications [11, 53].

The resampling stage has a significant impact to the computational burden of the PF algorithm. All the mentioned methods are \( O(N_p) \). However, the real impact depends on how often resampling is invoked. By using the SIS algorithm, this happens at any time step \( k \). Therefore, the total complexity of PF is \( O(N_p^2 n^2) \). If the SIR algorithm is used, the impact of resampling depends on the chosen threshold \( N_{th} \), on the frequency of particles degeneration, and on the used importance distribution \( q \).

3.5.4 PF for Planar Tracking

The use of PFs for planar tracking has been widely investigated during last two decades (e.g. [11, 53, 95]). For applying Algorithm 3.4, the following elements have to be determined:

- The initial state PDF \( p(x_0) \): by assumption (3.a): \( x_0 \sim p(x_0) = N(\bar{x}_0, \Psi_{x_0}) \).

- The state conditional PDF \( p(x(k+1)|x(k)) \): model (3.3) is linear with additive noise, usually assumed to be Gaussian, therefore:

\[
p(x(k+1)|x(k)) = N(Ax(k), F\Psi_w F^T)
\]

- The measurement conditional PDF \( p(y(k)|x(k)) \): model (3.4) is nonlinear but with additive noise, usually supposed to be Gaussian. The measurement function can be rewritten
as
\[ y(k) = \tilde{h}(x(k)) + v(k) \tag{3.26} \]
where
\[ \tilde{h}(x) := \begin{bmatrix} \sqrt{x_1^2 + x_4^2} \\ \arctan \left( \frac{x_4}{x_1} \right) \end{bmatrix} \]
from which:
\[ p(y(k)|x(k)) = N \left( \tilde{h}(x(k)), \Psi_v \right). \]

- The importance distribution \( q \) (line 10-11): the usual choice is \( q(x^{(i)}_{k+1}|x^{(i)}_k, Y_k) = p(x^{(i)}_{k+1}|x^{(i)}_k), \) which simplifies line 11 into \( \omega^{(i)}_{k+1|k} = \omega^{(i)}_{k|k}. \)

- The resampling strategy: SIS (Algorithm 3.6) is the most used, with \( N_{th} = \frac{2}{3} N_p. \)

- The number of particles \( N_p: \) this can be considered a tuning parameter. As widely discussed, the estimate accuracy is improved as larger is \( N_p. \) On the other hand, the computational burden linearly grows with \( N_p \) (without considering the resampling step). Therefore, setting \( N_p \) is of fundamental importance since it regulates the trade-off between accuracy and complexity. Usually, thousands of particles are required to get acceptable performances.

Finally, it is worth mentioning that a tuning strategy similar to the one introduced for EKF, with parameters \( \eta_w \) and \( \eta_v \) may be required (see Section 3.3.1). In particular setting a large amount of state noise (i.e. \( \eta_w \gg 1 \)) often is crucial to obtain acceptable results and prevent the degeneration of particles.
LVMKF and $\nu$-PVMKF for the 2-DTP

The principal contributions of this dissertation are illustrated in this chapter. The basic idea is to solve the 2-DTP by using the virtual measurement approach, which is described in Section 1.3. For clarity, the main concepts are here recalled.

A stochastic discrete-time dynamical system having the form:

$$x(k + 1) = Ax(k) + Bu(k) + Fw(k),$$
$$y(k) = h(x(k), u(k), v(k))$$

(4.1) (4.2)

is said to admit a virtual measurement map if there exist a completely observed sequence $\{y_v(k)\}$, named virtual output, a matrix $C(k)$, and a random white sequence $\{n(k)\}$, such that, for any $k \geq 0$, one has:

$$y_v(k) = C(k)x(k) + n(k).$$

(4.3)

A linear stochastic discrete-time system endowed with a virtual measurement map is referred to as virtual linear stochastic model.

A virtual linear stochastic model has a form amenable for applying the standard linear KF algorithm. The idea of this work is to prove that the tracking model (1.1), (1.3), and (1.4) admits a virtual measurement map and design a filtering algorithm by applying KF to the transformed system. The so obtained procedure has the major advantage of avoiding any linearization. Moreover, the simplicity and computational complexity of KF are preserved. The only drawback is due to the nongaussianity of the transformed noise $n(k)$. In fact, the virtual
measurement transforms nonlinearity into nongaussianity, even if the original measurement noise is Gaussian. The idea here is to prefer to avoid linearization and accept to use an estimator which is suboptimal because of nongaussianity. However, a second algorithm is further proposed in order to treat the transformed noise with more efficiency. Such an improvement is obtained by using a polynomial filtering technique, which is based on the theory developed during last decades by [18, 19, 35, 49].

In the following section, an optimality criterion for the filtering of systems admitting a virtual measurement map is established, yielding to two algorithms named Linear Virtual Measurement Kalman Filter (LVMKF) and $\nu$-Polynomial Virtual Measurement Kalman Filter ($\nu$-PVMKF), respectively. The subsequent sections illustrate how to get a LVMKF and a $\nu$-PVMKF for solving the 2-DTP.

4.1 Linear and $\nu$-Polynomial Virtual Measurement Kalman Filters

Equations (4.1) and (4.3) models a linear nongaussian discrete-time system. As discussed in Section 3.2, because of nongaussianity, no optimal estimator can be designed for such a class of systems. However, a form of sub-optimality can be defined by using the geometrical approach proposed in Section 3.2.

Recall that the optimal filtering solution can be obtained by the projection of the system state onto the space of all possible Borel transformations of the output sequence \( \{y_v(\tau), \tau = 0, 1, \ldots, k\} \) (see (3.6)), i.e.

\[
\hat{x}(k) = E[x(k)|Y_k^v] = \Pi(x(k)|L^2(Y_k^v, n)),
\]

where \( Y_k^v \) is a \((k + 1)q\) random vector:

\[
Y_k^v := \begin{bmatrix} y_v(0) \\ y_v(1) \\ \vdots \\ y_v(k) \end{bmatrix}.
\]

Such a projection cannot be computed with a finite number of parameters when noise is nongaussian. A possible suboptimal solution is to look for the best affine transformation of \( Y_k^v \), in terms of minimum variance, by projecting the system state onto the subspace \( \mathcal{L}(Y_k^v, n) \subset L^2(Y_k^v, n) \), which is defined in (3.9). Therefore,

\[
\hat{x}_{LVME}(k) := \Pi(x(k)|\mathcal{L}(Y_k^v, n))
\]

is the minimum variance linear estimate with respect to the virtual measurement \( y_v \), named linear virtual measurement estimate (LVME). From Proposition 3.2 follows that \( \hat{x}_{LVME}(k) \) can be computed by directly applying KF to the virtual linear stochastic model. This yields the best minimum variance linear filtering algorithm with respect to \( y_v \), denoted as Linear Virtual Measurement Kalman Filter (LVMKF).

In [18] the idea of enlarging the subspace into which the state is projected is introduced to improve the effectiveness of Kalman-like algorithms for linear nongaussian discrete-time stochastic systems. In particular, the subspace of all polynomial transformation of \( Y_k^v \) of order
4.1. Linear and $\nu$-Polynomial Virtual Measurement Kalman Filters

Figure 4.1: The polynomial filtering approach: the optimal estimate is the orthogonal projection of the state $x$ onto the space $L^2$ (dotted line). The suboptimal estimates are: the projection $\hat{x}_L$ onto the space of linear transformations of the output $L$ (dot-dashed); the projection $\hat{x}_2$ onto the space of 2-nd order polynomial transformations of the output $P^2$ (short-dashed); the projection $\hat{x}_3$ onto the space of 3-rd order polynomial transformations of the output $P^3$ (long-dashed). As the polynomial order increases the distance (i.e. the error variance) becomes smaller.

$\nu > 0$ is defined as

$$\mathcal{P}^\nu(Y^v_k, n) := \left\{ z : \Omega \to \mathbb{R}^n \mid \exists p^\nu : \mathbb{R}^{(k+1)q} \to \mathbb{R}^n \text{ s.t. } z = p^\nu(Y^v_k) \right\} \subset L^2(Y^v_k, n),$$

where $p^\nu(\cdot)$ is a polynomial function of order $\nu$. It is clear that $\mathcal{P}^1(Y^v_k, n) \equiv L(Y^v_k, n)$ and

$$\mathcal{P}^1(Y^v_k, n) \subset \mathcal{P}^2(Y^v_k, n) \subset \cdots \subset \mathcal{P}^\nu(Y^v_k, n) \subset L^2(Y^v_k, n).$$

Therefore, projecting the state onto $\mathcal{P}^\nu(Y^v_k, n)$ will return an estimate having an error variance (i.e. the distance in $L^2(Y^v_k, n)$) equal or smaller than that of the linear estimate, yielding to a possible improvement in terms of accuracy. Figure 4.1 graphically illustrates this idea. The minimum variance polynomial estimate, of order $\nu$, with respect to the virtual measurement $y_v$ is hence defined as

$$\hat{x}_{\nu-PVME}(k) := \Pi(x(k) | \mathcal{P}^\nu(Y^v_k, n))$$

and referred to as the $\nu$-polynomial virtual measurement estimate ($\nu$-PVME); moreover, a recursive algorithm able to return the $\nu$-PVME will be the best minimum variance polynomial filtering algorithm, of order $\nu$, with respect to $y_v$, denoted as $\nu$-Polynomial Virtual Measurement Kalman Filter ($\nu$-LVMKF). This solution can be considered as intermediate among the conventional linear filtering and the infinite dimensional optimal filtering.

Next two sections are dedicated to develop a LVMKF and a $\nu$-PVMKF for solving the 2-DTP.
4.2 LVMKF for the 2-DTP

The following theorem states that the tracking model (1.1), (1.3), and (1.4) is endowed with a virtual measurement map.

**Theorem 4.1.** The tracking system defined by (1.1), (1.3), and (1.4) admits a virtual measurement map with:

\[ y_v(k) = R^T(\theta_m(k)) \begin{bmatrix} \rho_m(k) \\ 0 \end{bmatrix}, \quad C(k) = C = \bar{R}S, \quad (4.6) \]

\[ n(k) = \hat{R}(k)x(k) + R^T(\theta_m(k))Gn_\rho(k), \quad (4.7) \]

where

\[ R(\alpha) := \begin{bmatrix} \cos(\alpha) & \sin(\alpha) \\ -\sin(\alpha) & \cos(\alpha) \end{bmatrix}, \quad \bar{R} := E[R(-v_\theta(k))] \quad (4.8) \]

\[ \hat{R}(k) := (R(-v_\theta(k)) - \bar{R})S, \quad (4.9) \]

\[ S := \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad G := \begin{bmatrix} 1 \\ 0 \end{bmatrix}. \quad (4.10) \]

The measurement error covariance matrix is given by

\[ \Psi_n(k) = \text{st}^{-1} \left( E \left[ \hat{R}(k)[2] \right] \text{st} \left( \Psi_x(k) \right) \right) + \psi^{(2)}_\rho R^T(\theta_m(k))GG^T R(\theta_m(k)), \quad (4.11) \]

with

\[ \Psi_x(k) = A\Psi_x(k - 1)A^T + F \text{st}^{-1} \left( \psi^{(2)}_w \right) F^T, \quad \Psi_x(0) = \Psi_{x0}, \quad (4.12) \]

where \( \text{st}(\cdot) \) and \( \text{st}^{-1}(\cdot) \) denote the stack operator and its inverse defined in Definition A.3.

**Proof.** Taking into account that

\[ x_1(k)\cos(\theta(k)) = \rho(k) \cos^2(\theta(k)) \]

\[ x_4(k)\sin(\theta(k)) = \rho(k) \sin^2(\theta(k)) \]

and the output equation (1.3), it readily follows that

\[ \rho_m(k) - v_\rho(k) = \rho(k) = \cos(\theta_m(k) - v_\theta(k))x_1(k) + \sin(\theta_m(k) - v_\theta(k))x_4(k). \quad (4.13) \]

Moreover, from (1.4):

\[ 0 = -\sin(\theta_m(k) - v_\theta(k))x_1(k) + \cos(\theta_m(k) - v_\theta(k))x_4(k). \quad (4.14) \]

Now, from (4.13) and (4.14) it results that

\[ \begin{bmatrix} \rho_m(k) \\ 0 \end{bmatrix} = R(\theta_m(k) - v_\theta(k))Sx(k) + Gv_\rho(k), \quad (4.15) \]
where $R(\cdot)$, $S$, and $G$ are the matrices defined in (4.8) and (4.10). Note that matrix $R(\cdot)$ satisfies the following properties for any $\alpha, \beta \in \mathbb{R}$:

\[
R(\alpha + \beta) = R(\alpha)R(\beta), \quad R^{-1}(\alpha) = R^T(\alpha) = R(-\alpha). \tag{4.16}
\]

By exploiting (4.16)-(4.17), (4.15) is rewritten in the form

\[
R^T (\theta_m(k)) \begin{bmatrix} \rho_m(k) \\ 0 \end{bmatrix} = R(-v_\theta(k)) Sx(t) + R^T (\theta_m(k)) Gv_\rho(k). \tag{4.18}
\]

The left-hand side of last equation corresponds to the virtual output $y_v(k)$ given in (4.6). Finally, through a simple manipulation, one has

\[
y_v(k) = \hat{R}Sx(t) + (R(-v_\theta(k)) - \hat{R}) Sx(t) + R^T (\theta_m(k)) Gv_\rho(k),
\]

which corresponds to the virtual measurement map given in (4.6) and (4.7).

It remains to prove that the random sequence $\{n(k)\}$ is zero mean, white and uncorrelated with $\{x(k)\}$. This readily derives by observing that $\{n(k)\}$ is a combination of the three independent random sequences: $\{\hat{R}(k)\}$, $\{v_\rho(k)\}$ and the state sequence $\{x(k)\}$, where the fist two are zero mean and white.

As far as (4.12) and (4.11) are concerned, by using (A.5) in Lemma A.4 and taking into account (1.5), one has:

\[
\Psi_x(k) := E \left[ x(k)x^T(k) \right] = E[\{(Ax(k-1) + Fa(k-1))(Ax(k-1) + Fa(k-1))^T\}] = A\Psi_x(k-1)A^T + Fst^{-1} \left( \psi_w^{(2)} \right) F^T,
\]

and, thanks to (A.4)-(A.5) in Lemma A.4 and the statistical properties of $\{\hat{R}(k)\}$, $\{v_\rho(k)\}$ and $\{x(k)\}$:

\[
\Psi_n(k) := E \left[ n(k)n^T(k) \right] = E \left[ \hat{R}(k)x(k)x^T(k) \hat{R}^T(k) \right] + \psi_\rho^{(2)} R^T (\theta_m(k)) GG^T R(\theta_m(k)) = st^{-1} \left( E \left[ \hat{R}(k)^2 \right] st \left( \Psi_x(k) \right) \right) + \psi_\rho^{(2)} R^T (\theta_m(k)) GG^T R(\theta_m(k)).
\]

Now it is possible to state the first main result of the thesis which readily follows from Theorem 4.1 and Proposition 3.2.

**Proposition 4.2.** The 2-DTP admits a virtual model representation

\[
x(k + 1) = Ax(k) + Fw(k) \tag{4.19}
\]

\[
y_v(k) = Cx(k) + n(k) \tag{4.20}
\]
4.2. LVMKF for the 2-DTP

1: **given**: $A$, $F$, $\psi_w^{(2)}$, $\psi_\rho^{(2)}$, $\rho_m(k)$, $\theta_m(k)$, $\forall k = 0, 1, \ldots, T$

2: Compute time-invariant quantities: $S$, $G$, $\bar{R}$ and $C = \bar{R}S$ by (4.10), (4.8) and (4.7), and the state covariance matrix $\Psi_w = st^{-1}\left(\psi_w^{(2)}\right)$

3: Initial state covariance matrix: $\Psi_x(0) = \Psi_{x0}$

4: Initial estimate: $\hat{x}(0|\bar{-1}) = \bar{x}_0$, $P(0|\bar{-1}) = \Psi_{x0}$

5: for $k = 0 \rightarrow T$ do

6: ▶ Correction

7: $\Psi_n(k) = st^{-1}\left(\mathbb{E}\left[R(k)^{[2]}\right]st(\Psi_x(k)) + \psi_\rho^{(2)}R^T(\theta_m(k))GG^T R(\theta_m(k))\right)$

8: $y_v(k) = R^T(\theta_m(k))\left[\begin{array}{c}
\rho_m(k) \\
0
\end{array}\right]^T$

9: $K(k) = P(k|k-1)C^T(k)\left(C(k)P(k|k-1)C^T(k) + \Psi_n(k)\right)^{-1}$

10: $\hat{x}(k|k) = \hat{x}(k|k-1) + K(k)(y_v(k) - C\hat{x}(k|k-1))$

11: $P(k|k) = (I_6 - K(k)C(k)) P(k|k-1)$

12: ▶ Prediction

13: $\hat{x}(k+1|k) = A\hat{x}(k|k)$

14: $P(k+1|k) = AP(k|k)A^T + F\Psi_w F^T$

15: $\Psi_x(k+1) = A\Psi_x(k)A^T + F\Psi_w F^T$

16: end for

Algorithm 4.1: Linear Virtual Measurement Kalman Filter for the 2-DTP

where $y_v(k)$, $C$ and $n(k)$ are given by (4.6)-(4.10) and, therefore, KF (Algorithm 3.1) applied to this model constitutes the LVMKF for the 2-DTP.

Algorithm 4.1 reports the procedure of the LVMKF for the 2-DTP. Notice that the computational complexity of this algorithm is the same of EKF (Algorithm 3.2), which is $O(n^3)$.

Two quantities in Algorithm 4.1 are not explicitly provided: $\bar{R}$ and $\mathbb{E}(\bar{R}^{[2]}(k))$. This is because, in general, no hypothesis on the distribution of $v_\theta(k)$ is given. Their computation must be carried out using the statistical moments $\psi_\theta^{(i)}$, supposed known by Assumption (1.d). More precisely, they can be computed as particular cases of (4.29) and (4.52) with $\nu = 1$. 
4.3 \( \nu \)-PVMKF for the 2-DTP

The aim of this section is to build a filter able to return the estimate of the state \( x(k) \) of the virtual model \((4.19)-(4.20)\) as projection onto the space of the polynomial transformations of \( y_v \), indicated in \((4.4)\) with \( \mathcal{P}^\nu (Y_v^k, n) \).

The first step for building the announced filter is to introduce the extended polynomial virtual measurement vector and the corresponding extended state:

**Definition 4.3.** For the system model \((4.19)-(4.20)\), the extended output \( Y_v(k) \) and the extended state \( X(k) \) are defined as

\[
Y_v(k) = \begin{bmatrix}
Y_v^{(1)}(k) \\
Y_v^{(2)}(k) \\
\vdots \\
Y_v^{(\nu)}(k)
\end{bmatrix} \in \mathbb{R}^Q, \quad X(k) = \begin{bmatrix}
x(k) \\
x^{[2]}(k) \\
\vdots \\
x^{[\nu]}(k)
\end{bmatrix} \in \mathbb{R}^N,
\]

where \( \nu \) is the filter order, \( Q = 2(2^\nu - 1), N = 6(6^\nu - 1)/5 \), and

\[
Y_v^{(i)}(k) = y_v^{[i]}(k) - R_T^{[i]}(\theta_m(k))G_v^{[i]} \psi_r^{(i)}.
\]

Notice that \( Y_v^{(i)}(k) \) \((i = 0, 1, \ldots, \nu)\) contains information regarding the Kronecker powers of \( y_v(k) \), up to the order \( \nu \). Polynomial filtering is indeed based on the definition of an extended model which is required be constituted by an output and a state vector composed by all Kronecker powers of the original vectors, up to the filter order \( \nu \). If such augmented model is amenable to the application of standard KF, this will naturally return the best minimum variance polynomial estimate for the considered system. In particular, Theorem 4.7, together with the statistical characterization given in Theorem 4.9, will show that the extended output and state vectors in \((4.21)\) are linearly related within a model amenable to the use of KF. Finally, Proposition 4.10 will formally state that applying KF to the augmented model provides the required \( \nu \)-PVME.

Before to present the announced result, it is necessary to introduce some useful definitions and lemmas.

**Definition 4.4.** Let \( \mathcal{M} \) be the set of all finite dimensional matrices, then the map \( \Lambda_\nu, \nu \geq 1 \), is defined as

\[
\Lambda_\nu(M) = \begin{bmatrix}
M^{[1]} & 0 & \cdots & 0 \\
0 & M^{[2]} & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & M^{[\nu]} 
\end{bmatrix}, \quad M \in \mathcal{M}.
\]

**Lemma 4.5.** Let \( \Pi^{(i)}_{\nu,n} \in \mathbb{R}^{n^\nu \times r} \), \( i, n, \nu \geq 1, r = n(n^\nu - 1)/(n - 1) \) be the matrix

\[
\Pi^{(i)}_{\nu,n} := \begin{bmatrix}
0_{n^i \times n} & \cdots & 0_{n^i \times n^{i-1}} & I_{n^i} & 0_{n^i \times n^{i+1}} & \cdots & 0_{n^i \times n^\nu}
\end{bmatrix},
\]
where $0_{p \times q}$ denotes the $p \times q$ null matrix. Given $M \in \mathbb{R}^{r \times s}$, $L \in \mathbb{R}^{s \times q}$, $N \in \mathbb{R}^{n \times n}$, $z \in \mathbb{R}^{n}$, and

$$Z = \begin{bmatrix} z & z^{[2]} & \cdots & z^{[\nu]} \end{bmatrix},$$

(4.24)

the following properties hold true:

$$\Lambda^{[i]}_{\nu}(ML) = \Lambda^{[i]}_{\nu}(M)\Lambda^{[i]}_{\nu}(L),$$

(4.25)

$$\Lambda_{\nu}(N) = \begin{bmatrix} N^{[1]}\Pi_{\nu,n}^{(1)} & 0 & \cdots & 0 \\ N^{[2]}\Pi_{\nu,n}^{(2)} & \cdots & \cdots & 0 \\ \vdots & \cdots & \cdots & \vdots \\ N^{[\nu]}\Pi_{\nu,n}^{(\nu)} & \cdots & \cdots & 0 \end{bmatrix},$$

(4.26)

$$z^{[i]} = \Pi_{\nu,n}^{(i)}Z.$$  

(4.27)

**Proof.** Property (4.25) can be readily proved by exploiting the diagonal structure of operator $\Lambda_{\nu}$ and noting that for any pair of suitable dimensioned matrices $M$ and $L$ it results that

$$(ML)^{[i]} = M^{[i]}L^{[i]},$$

which can be proved by iteratively applying property (A.2) in Lemma A.4. Properties (4.26) and (4.27) can be proved by direct computation. \(\blacksquare\)

**Lemma 4.6.** Let $V_{\rho}(k) \in \mathbb{R}^\nu$ be defined as

$$V_{\rho}(k) = \begin{bmatrix} v_{\rho}(k) & v_{\rho}^{2}(k) & \cdots & v_{\rho}^{\nu}(k) \end{bmatrix}^T \in \mathbb{R}^\nu.$$  

Then:

$$\bar{V}_{\rho} := E[V_{\rho}(k)] = \begin{bmatrix} \psi_{\rho}^{(1)} & \psi_{\rho}^{(2)} & \cdots & \psi_{\rho}^{(\nu)} \end{bmatrix},$$

(4.28)

$$\bar{R}^{(i)} := E[R^{[i]}(-v_{\theta}(k))] = T^{[i]}L_{i}(T^{-1})^{[i]},$$

(4.29)

$$\bar{\Lambda} := E[\Lambda_{\nu}(R(-v_{\theta}(k))S)] = \begin{bmatrix} \bar{R}^{(1)}S & 0 & \cdots & 0 \\ 0 & \bar{R}^{(2)}S & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \bar{R}^{(\nu)}S \end{bmatrix},$$

(4.30)

with

$$T = \begin{bmatrix} -j & j \\ 1 & 1 \end{bmatrix}, \quad L_{i} = \text{diag}\left(E[e^{jL_{i}(q)v_{\theta}(k)}], q = 1, 2, \ldots, 2^{i}\right),$$

(4.31)
where \( j \) denotes the imaginary unit, and \( l_i(q) \in \mathbb{Z} \) satisfies
\[
l_i(q) = 2 \sum_{m=1}^{i-1} \left\| \frac{q - 1}{2^{i-m}} \right\|_2 + 2|q - 1|_2 - i, \tag{4.32}
\]
in which \( \lfloor \cdot \rfloor \) and \( |\cdot|_2 \) denote integer part and 2-modulo, respectively.

**Proof.** As far as \( \bar{V}_p \) is concerned, it is enough to note that such a vector is composed by all the moments of the zero mean random variable \( z_{\rho}(k) \) up to the order \( \nu \), indicated as defined in (1.5).

Regarding \( \bar{R}^{(i)} \), it is easy to verify that the eigenvalues of matrix \( R^{(\cdot)} \), defined in (4.8), are
\[
\lambda_1(k) = e^{-j\nu_{\theta}(k)}, \quad \lambda_2(k) = e^{j\nu_{\theta}(k)},
\]
and that the diagonalizing matrix \( T \), constituted by the corresponding eigenvectors, is that given by (4.31). Therefore, (4.29) holds true by defining \( L_i \) as
\[
L_i := E\left[ diag \left( z^{[i]}(k) \right) \right], \text{ with } z(k) = \begin{bmatrix} \lambda_1(k) & \lambda_2(k) \end{bmatrix}^T. \tag{4.33}
\]
From Lemma A.9, it results that the \( q \)-th entry of \( z^{[i]}(k) \), satisfies
\[
\left( z^{[i]}(k) \right)_q = z_{l_1}(k)z_{l_2}(k) \cdots z_{l_i}(k), \tag{4.34}
\]
where coefficients \( l_m \in \{1, 2\} \) \( (m = 1, 2, \ldots, i) \) are given by (A.9) in Lemma A.9. By taking into account (4.34), it follows that
\[
\left( z^{[i]}(k) \right)_q = \exp \left\{ j\nu_{\theta}(k) \left[ p_{i,2}(q) - p_{i,1}(q) \right] \right\}, \tag{4.35}
\]
where \( p_{i,1}(q) \) and \( p_{i,2}(q) \) denote the cardinalities of the sets of the all coefficients \( l_m \) equal to 1 and 2, respectively. From (A.9), it results that
\[
p_{i,2}(q) = \sum_{m=1}^{i} (l_m - 1) = \sum_{m=1}^{i-1} \left\| \frac{q - 1}{2^{i-m}} \right\|_2 + |q - 1|_2,
\]
\[
p_{i,1}(q) = i - p_{2}(q),
\]
from which, one has:
\[
l_i(q) := p_{i,1}(q) - p_{i,1}(q) = 2 \sum_{m=1}^{i-1} \left\| \frac{q - 1}{2^{i-m}} \right\|_2 + 2|q - 1|_2 - i. \tag{4.36}
\]
Finally, from (4.35) and (4.36), it follows that
\[
\left( E\left[ z^{[i]}(k) \right] \right)_q = E\left[ e^{j l_i(q) \nu_{\theta}(k)} \right],
\]
which is equal to (4.32).
Theorem 4.7. The extended output and extended state defined in (4.21) for the system model (1.1), (1.3) and (1.4), satisfy the following model equations:

\[
\mathcal{X}(k+1) = \mathcal{A} \mathcal{X}(k) + \mathcal{U} + \mathcal{W}(k),
\]

\[
\mathcal{Y}_{\nu}(k) = \mathcal{C}(k) \mathcal{X}(k) + \mathcal{N}(k),
\]

where

\[
\mathcal{A} = \begin{bmatrix}
    A & 0 & \ldots & 0 \\
    \mathcal{O}_{2,1} & A[2] & \ldots & 0 \\
    \vdots & \vdots & \ddots & \vdots \\
    \mathcal{O}_{\nu,1} & \mathcal{O}_{\nu,2} & \ldots & A[\nu]
\end{bmatrix},
\]

\[
\mathcal{C}(k) = \begin{bmatrix}
    C^{(1)}(\theta_m(k)) \\
    C^{(2)}(\theta_m(k)) \\
    \vdots \\
    C^{(\nu)}(\theta_m(k))
\end{bmatrix},
\]

\[
\mathcal{N}(k) = \begin{bmatrix}
    \mathcal{N}^{(1)}(k) \\
    \mathcal{N}^{(2)}(k) \\
    \vdots \\
    \mathcal{N}^{(\nu)}(k)
\end{bmatrix},
\]

\[
\mathcal{U}^{(1)}(k) = \begin{bmatrix}
    0 \\
    F^{[2]}(\psi_w^{(2)}) \\
    \vdots \\
    F^{[\nu]}(\psi_w^{(\nu)})
\end{bmatrix},
\]

\[
\mathcal{W}^{(1)}(k) = \begin{bmatrix}
    \mathcal{W}^{(1)}(k) \\
    \mathcal{W}^{(2)}(k) \\
    \vdots \\
    \mathcal{W}^{(\nu)}(k)
\end{bmatrix},
\]

\[
\mathcal{W}(k) = \begin{bmatrix}
    \mathcal{W}^{(1)}(k) \\
    \mathcal{W}^{(2)}(k) \\
    \vdots \\
    \mathcal{W}^{(\nu)}(k)
\end{bmatrix},
\]

\[
\mathcal{W}^{(i)}(k) = \sum_{l=0}^{i-1} M_{i-l,6}^{i} \left( F^{[i-l]} \otimes A[l] \right) \left( \psi_w^{(i-l)} \otimes I_{n[l]} \right),
\]

\[
\mathcal{C}^{(i)}(\theta_m(k)) = \mathcal{M}^{(i)}_{\nu,2} \left( \tilde{\Lambda} \otimes \Gamma(\theta_m(k)) \right) \left( I_N \otimes \hat{V}_\rho \right) + \bar{R}^{(i)} S^{[i]} \Pi^{(i)}_{\nu,6},
\]

\[
\mathcal{N}^{(i)}(k) = \mathcal{M}^{(i)}_{\nu,2} \left( \tilde{\Lambda} \otimes \Gamma(\theta_m(k)) \right) \left( \mathcal{X}(k) \otimes \hat{V}_\rho \right) + \mathcal{M}^{(i)}_{\nu,2} \left( \tilde{\Lambda} \otimes \Gamma(\theta_m(k)) \right) \left( \mathcal{X}(k) \otimes \hat{V}_\rho \right)\hat{\rho}^{(i)}(k) S^{[i]} \Pi^{(i)}_{\nu,6} \mathcal{X}(k) + R^{T[i]}(\theta_m(k)) G^{[i]} \Pi^{(i)}_{\nu,1} \hat{\rho}_\rho(k).
\]

in which $M_{k,n}^{i}$ and $\mathcal{M}_{\nu,n}^{(i)}$ are defined in Appendix by (A.7) and (A.13), respectively, and

\[
\Lambda(k) := \Lambda_\nu \left( R(-v_\theta(k)) S \right),
\]

\[
\hat{V}_\rho(k) := V_\rho(k) - \hat{V}_\rho \in \mathbb{R}^\nu,
\]

\[
\hat{\Lambda}(k) := \Lambda(k) - \tilde{\Lambda} \in \mathbb{R}^{Q \times N},
\]

\[
\tilde{\hat{R}}^{(i)}(k) := R^{[i]}(-v_\theta(k)) - \tilde{R}^{(i)} \in \mathbb{R}^{2^i \times 2^i},
\]

\[
\Gamma(\theta_m(k)) := \Lambda_\nu \left( R^T(\theta_m(k)) G \right) \in \mathbb{R}^{Q \times N}.
\]

Moreover, \{\mathcal{W}(k)\} and \{\mathcal{N}(k)\} are uncorrelated zero mean white sequences and the second is also uncorrelated with \{\mathcal{X}(k)\}. 

As far as $\bar{\Lambda}$ is concerned, (4.30) trivially follows from definitions in (4.23) and (4.29). ■
4.3. $\nu$-PVMKF for the 2-DTP

By applying Lemma A.10 to (4.22), and taking into account (4.49) and (4.44), it results:

$$y_v(k) = R(-v_\theta(k))Sx(k) + R^T(\theta_m(k))Gv_\rho(k). \tag{4.49}$$

By applying Lemma A.10 to (4.22), and taking into account (4.49) and (4.44), it results:

$$\mathcal{Y}_{\nu}^{(i)}(k) = \mathcal{M}_{\nu,2}^{(i)} \left( \Lambda(k) \otimes \Lambda_v \left( \mathcal{R}^T(\theta_m(k)) \mathcal{G} \right) \left( \mathcal{X} \otimes \hat{V}_\rho \right) \right)
+ R^{[i]}(-v_\theta(k))S^{[i]}x^{[i]}(k) + R^{T[i]}(\theta_m(k))G^{[i]}v^{[i]}_\rho(k)
- R^{T[i]}(\theta_m(k))G^{[i]}\psi^{(i)}_\rho(k).$$

Moreover, using definitions (4.45)-(4.48) and properties in Lemma A.4 and in (4.27), one has:

$$\mathcal{Y}_{\nu}^{(i)}(k) = \mathcal{M}_{\nu,2}^{(i)} \left( \Lambda \otimes \Gamma(\theta_m(k)) \right) \left( I_N \otimes \hat{V}_\rho \right) \mathcal{X}(k)
+ \mathcal{M}_{\nu,2}^{(i)} \left( \Lambda \otimes \Gamma(\theta_m(k)) \right) \left( \mathcal{X} \otimes \hat{V}_\rho \right)
+ \mathcal{M}_{\nu,2}^{(i)} \left( \Lambda \otimes \Gamma(\theta_m(k)) \right) \left( \mathcal{X} \otimes \hat{V}_\rho \right)
+ \mathcal{R}^{(i)}(k) S^{[i]} \Pi^{(i)}_{\nu,6} \mathcal{X}(k) + \mathcal{R}^{(i)}(k) S^{[i]} \Pi^{(i)}_{\nu,6} \mathcal{X}(k)
+ R^{T[i]}(\theta_m(k))G^{[i]} \Pi^{(i)}_{\nu,1} \hat{V}_\rho(k) + R^{T[i]}(\theta_m(k))G^{[i]} \Pi^{(i)}_{\nu,1} \hat{V}_\rho(k)
- R^{T[i]}(\theta_m(k))G^{[i]} \psi^{(i)}_\rho
= \mathcal{C}^{(i)}(\theta_m(k)) \mathcal{X}(k) + \mathcal{N}^{(i)}(k)$$

which proves (4.37).

To finish the proof, the whiteness of $\{\mathcal{W}(k)\}$ is demonstrated in [18], meanwhile the whiteness of $\{\mathcal{N}(k)\}$ and its uncorrelation with $\{\mathcal{X}(k)\}$ are readily proved by observing that: $\{\Lambda(k)\}$, $\{\mathcal{X}(k)\}$ and $\{\hat{V}_\rho(k)\}$ are independent; $\{\mathcal{R}^{(i)}(k)\}$ is independent of $\{\mathcal{X}(k)\}$ and $\{\hat{V}_\rho(k)\}$; and $\{\hat{\Lambda}(k)\}$ and $\{\hat{V}_\rho(k)\}$ are zero mean and white. Moreover, $\{\mathcal{W}(k)\}$ and $\{\mathcal{N}(k)\}$ are uncorrelated because of the independence of the state noise $\{w(k)\}$ and the measurement noises $\{v_\rho(k)\}$ and $\{v_\theta(k)\}$, as stated in the problem formulation.

In order to build the optimal linear filter for the extended model (4.37) it is necessary to verify that the extended state and output noises admit finite covariances. Before to provide such matrices it is useful to characterize the statistics of the random sequences $\{\hat{V}_\rho(k)\}$ and $\{\hat{\Lambda}(k)\}$ through the following Lemma.

**Lemma 4.8.** The second order statistical matrices $\Psi_\rho \in \mathbb{R}^{\nu \times \nu}$ and $\Psi_\theta \in \mathbb{R}^{N^2 \times N^2}$ defined as

$$\Psi_\rho = E \left[ \hat{V}_\rho(k) \hat{V}_\rho^T(k) \right] \tag{4.50}$$
$$\Psi_\theta = E \left[ \hat{\Lambda}^{[2]}(k) \right], \tag{4.51}$$

satisfy:

$$(\Psi_\rho)_{r,s} = \psi^{(r+s)} - \psi^{(r)} \psi^{(s)}$$
\[ \Psi_{\theta} = \Lambda_{\nu}^{[2]}(T)\bar{\Lambda}^{(2)}\Lambda_{\nu}^{[2]}(T^{-1}S) - \bar{\Lambda}^{[2]} \]  

(4.52)

where \( \psi_{\rho}^{(i)} \), \( T \) are defined in (1.5) and (4.31) and \( \bar{\Lambda}^{(2)} \) is given by:

\[ \bar{\Lambda}^{(2)} = \text{diag} \left[ \sum_{h_1,\ldots,h_\nu \geq 0} M_{h_1,\ldots,h_\nu}^2 \left[ \left( \bigotimes_{i=1}^{\nu} \left( \Pi_{\nu,2}^{(i)} \right)^{h_1} \right) \text{diag}(L_{h_1+2h_2+\ldots+\nu}) \right] \right] \]

(4.53)

with \( L_i \) defined in (4.31).

**Proof.** As far as \( \Psi_{\rho} \) is concerned, the \((r,s)\) entry is given by

\[
(\Psi_{\rho})_{r,s} = E \left[ \begin{pmatrix} \hat{V}_{\rho}(k) \\ \hat{V}_{\rho}(k) \end{pmatrix}_r \begin{pmatrix} \hat{V}_{\rho}(k) \\ \hat{V}_{\rho}(k) \end{pmatrix}_s \right]
= E \left[ \begin{pmatrix} v_{\rho}^r(k) - \psi_{\rho}^{(r)} \\ v_{\rho}^s(k) - \psi_{\rho}^{(s)} \end{pmatrix} \right] = \psi_{\rho}^{(r+s)} - \psi_{\rho}^{(r)}\psi_{\rho}^{(s)}. \]

Recalling that \( T \) diagonalizes matrix \( R(-v_{\theta}(k)) \) and by exploiting property (4.25), it results

\[
E \left[ \Lambda_{\nu}^{[2]}(R(-v_{\theta}(k))S) \right] = E \left[ \Lambda_{\nu}^{[2]}(T\text{diag}(z(k))T^{-1}S) \right]
= \Lambda_{\nu}^{[2]}(T) E \left[ \Lambda_{\nu}^{[2]}(\text{diag}(z(k))) \right] \Lambda_{\nu}^{[2]}(T^{-1}S),
\]

(4.54)

where \( z(k) \) is the same vector in (4.33); moreover, by observing that

\[
\Lambda_{\nu}^{[2]}(\text{diag}(z(k))) = \text{diag}(Z_{[2]}(k)), \text{ with } Z(k) = \begin{bmatrix} z(k) \\ z_{[2]}(k) \\ \vdots \\ z_{[\nu]}(k) \end{bmatrix},
\]

Lemma A.8 implies that

\[
\bar{\Lambda}^{(2)} := E \left[ \Lambda_{\nu}^{[2]}(\text{diag}(z(k))) \right] = E \left[ \text{diag}(Z_{[2]}(k)) \right]
\]

(4.55)

satisfies (4.53) since \( L_i = E[\text{diag}(z^{[i]}(k))] \), as defined in (4.33). Finally, by recalling definitions (4.30), (4.44), and (4.46), one has:

\[
E \left[ \bar{\Lambda}^{[2]}(k) \right] = E \left[ \Lambda_{\nu}^{[2]}(R(-v_{\theta}(k))S) \right] - \bar{\Lambda}^{[2]}
\]

that can be readily proved to correspond to (4.51) by substituting (4.54) to the first term and taking into account (4.55). ■

**Theorem 4.9.** The covariances of \( \{W(k)\} \) and \( \{N(k)\} \) are given by:

\[
\Psi_{W}(k) := E \left[ W(k)W^T(k) \right], \Psi_{N}(k) := E \left[ N(k)N^T(k) \right]
\]

(4.56)
where
\[
(\Psi_W)_{r,s}(k) = \sum_{l=0}^{r-1} \sum_{m=0}^{s-1} M_{r-l,0}^T(F^{l-r} \otimes A^{l}) P_{l,m}^{r,s}(k)(F^{s-m} \otimes A^{m})(M_{s-m,0}^s)^T
\]  
(4.57)
with
\[
P_{l,m}^{r,s}(k) = st^{-1}
\left[
(I_2^{[s-m]} \otimes C_{2^{r-l,0}}^{T} \otimes I_6^{[l]})
\cdot \left((\psi_w^{(s-r+m)} - \psi_w^{(s-r-l)} \otimes C_{1,6}^{m} \otimes I_6^{[l]}) \Pi_{\nu,6}^{(l+m)} \mu_X(k) \right)
\right],
\]  
(4.58)
and where
\[
\Psi_N(k) = \sum_{\ell=1}^{4} \sum_{m=\ell+1}^{4} (\Phi_{\ell,m}(k) + \Phi_{\ell,m}^T(k)) + \sum_{\ell=1}^{4} \Phi_{\ell,\ell}(k),
\]  
(4.59)
with the non-zero terms satisfying:
\[
\Phi_{1,1}(k) = \mathcal{M}_{\nu,2} \left(st^{-1} \left( (\Psi_0 + \bar{A}^2) st(\Psi_X(k)) \right) \otimes \Xi(k) \right) \mathcal{M}_{\nu,2}^T,
\]
\[
\Phi_{1,4}(k) = \mathcal{M}_{\nu,2} \left((\bar{\Lambda} \mu_X(k)) \otimes \Xi(k) \right),
\]
\[
\Phi_{2,2}(k) = \mathcal{M}_{\nu,2} (\Theta(k) \otimes (\Gamma(\theta_m(k)) \bar{V}_{\rho} \bar{V}_{\rho}^T \Gamma^T(\theta_m(k)))) \mathcal{M}_{\nu,2}^T,
\]
\[
\Phi_{2,3}(k) = \mathcal{M}_{\nu,2} (\Theta(k) \otimes (\Gamma(\theta_m(k)) \bar{V}_{\rho})),
\]
\[
\Phi_{3,3}(k) = \Theta(k),
\]
\[
\Phi_{4,4}(k) = \Xi(k),
\]  
(4.60)
in which
\[
\Theta(k) = st^{-1} (\Psi_0 st(\Psi_X(k))),
\]
\[
\Xi(k) = \Gamma(\theta_m(k)) \Psi_\rho \Gamma^T(\theta_m(k)),
\]  
(4.61)
\[
\mathcal{M}_{\nu,2} = \left[
\mathcal{M}_{\nu,2}^{(1)}^T \ \mathcal{M}_{\nu,2}^{(2)}^T \ \ldots \ \mathcal{M}_{\nu,2}^{(\nu)}^T
\right] \in \mathbb{R}^{Q \times Q^2},
\]
\[
\mu_X(k) := E[\mathcal{X}(k)] = \mathcal{A} \mu_X(k-1) + \mathcal{U},
\]
\[
\Psi_X(k) := E[\mathcal{X}(k) \mathcal{X}^T(k)]
\]
\[
= \mathcal{A} \Psi_X(k-1) \mathcal{A}^T + \Psi_W(k-1)
\]
\[
+ \mu_X(k-1) \mathcal{U} + \mathcal{U} \mu_X^T(k-1).
\]  
(4.62)

**Proof.** Concerning \(\Psi_W(k)\) one can refer to [18].
From the expression of the components of $\mathcal{N}(k)$, given in (4.43), and using property (4.26), it can be noted that the whole $\mathcal{N}(k)$ consists of the sum of the following terms:

\[ \mathcal{L}_1(k) = \mathcal{M}_{\nu,2} \left( \Lambda(k) \otimes \Gamma(\theta_m(k)) \right) \left( \mathcal{X}(k) \otimes \hat{V}_\rho(k) \right), \]

\[ \mathcal{L}_2(k) = \mathcal{M}_{\nu,2} \left( \hat{\Lambda}(k) \otimes \Gamma(\theta_m(k)) \right) \left( \mathcal{X}(k) \otimes \hat{V}_\rho \right), \]

\[ \mathcal{L}_3(k) = \hat{\Lambda}(k) \mathcal{X}(k), \]

\[ \mathcal{L}_4(k) = \Gamma(\theta_m(k)) \hat{V}_\rho(k). \]

Equation (4.59) holds true by introducing the notation

\[ \Phi_{\ell,m}(k) = E \left[ \mathcal{L}_\ell(k) \mathcal{L}_m^T(k) \right], \quad \ell, m = 1, \ldots, 4. \]

In order to prove expression (4.60), considering $\Phi_{1,1}(k)$, one has:

\[ \Phi_{1,1}(k) = \mathcal{M}_{\nu,2} \tilde{\Phi}_{1,1}(k) \mathcal{M}_{\nu,2}^T, \]

with

\[ \tilde{\Phi}_{1,1}(k) = E \left[ \left( \Lambda(k) \otimes \Gamma(\theta_m(k)) \right) \left( \mathcal{X}(k) \otimes \hat{V}_\rho(k) \right) \left( \Lambda(k) \otimes \Gamma(\theta_m(k)) \right) \mathcal{X}(k) \otimes \hat{V}_\rho(k) \right]^T \]

\[ = E \left[ \left( \Lambda(k) \otimes \Gamma(\theta_m(k)) \right) \mathcal{X}(k) \otimes \hat{V}_\rho(k) \right] \left( \Lambda(k) \otimes \Gamma(\theta_m(k)) \right) \mathcal{X}(k) \otimes \hat{V}_\rho(k) \]

\[ = E \left[ \left( \Lambda(k) \mathcal{X}(k) \right) \otimes \left( \Gamma(\theta_m(k)) \right) \mathcal{X}(k) \otimes \hat{V}_\rho(k) \right] \]

\[ = \mathcal{M}_{\nu,2} \left( \left( \Psi + \tilde{\Lambda}[2] \right) \mathcal{M}_{\nu,2}^T \right) \mathcal{M}_{\nu,2}^T. \]

where the independence of $\{\Lambda(k)\}$, $\{\mathcal{X}(k)\}$, and $\{\hat{V}_\rho(k)\}$, and definitions (4.50) and (4.51) have been taken into account. All the other terms of (4.59) can be obtained following the same line. It will only result useful to rewrite in advance $\mathcal{L}_3(k)$ and $\mathcal{L}_4(k)$ in the form

\[ \mathcal{L}_3(k) = \left( \hat{\Lambda}(k) \otimes 1 \right) \left( \mathcal{X}(k) \otimes 1 \right), \]

\[ \mathcal{L}_4(k) = (1 \otimes \Gamma(\theta_m(k))) \left( 1 \otimes \hat{V}_\rho(k) \right). \]

As far as (4.62) and (4.63) are concerned, one has:

\[ \mu_{\mathcal{X}}(k) = E[A \mathcal{X}(k - 1) + U + \mathcal{W}(k)] = A \mu_{\mathcal{X}}(k - 1) + U, \]

\[ \Psi_{\mathcal{X}}(k) = E \left[ (A \mathcal{X}(k - 1) + U + \mathcal{W}(k)) (A \mathcal{X}(k - 1) + U + \mathcal{W}(k))^T \right] \]

\[ = A \Psi_{\mathcal{X}}(k - 1) A^T + \Psi_{\mathcal{W}}(k - 1) + \mu_{\mathcal{X}}(k - 1) U^T + U \mu_{\mathcal{X}}(k - 1). \]
Finally, it is possible to state the second main result of this thesis.

**Proposition 4.10.** KF applied to the extended state model (4.37) constitutes the $\nu$-PVMKF for the 2-DTP.

**Proof.** By Theorems 4.7 and 4.9, the extended state model (4.37) is amenable to the application of KF. By Proposition 3.2, this will provide the estimate

$$\hat{X}(k|k) = \Pi(\mathcal{X}(k)|\mathcal{L}(\mathcal{Y}_k^\nu, N))$$

where $\mathcal{Y}_k^\nu$ is the $(k+1)Q$ vector

$$\mathcal{Y}_k^\nu := \begin{bmatrix} \mathcal{Y}_v(0) \\ \mathcal{Y}_v(1) \\ \vdots \\ \mathcal{Y}_v(k) \end{bmatrix}.$$

Space $\mathcal{L}(\mathcal{Y}_k^\nu, N)$ has the following form:

$$\mathcal{L}(\mathcal{Y}_k^\nu, N) = \left\{ z : \Omega \rightarrow \mathbb{R}^N \mid \exists M \in \mathbb{R}^{N \times (k+1)Q}, \bar{m} \in \mathbb{R}^N \text{ s.t. } z = \bar{m} + M\mathcal{Y}_k^\nu \right\},$$

where

$$z = \bar{m} + M\mathcal{Y}_k^\nu = \bar{m} + \sum_{\tau=0}^{k} M_\tau \mathcal{Y}_v(\tau),$$

with $M_\tau, \tau = 0, 1, \ldots, k$ being the $N \times Q$ sub-matrices of $M$. Then, taking into account (4.21) and (4.22), it results that

$$z = \bar{m} + M\mathcal{Y}_k^\nu = \bar{m} - \sum_{\tau=0}^{k} \sum_{i=0}^{\nu} M_{\tau,i} R T^{[i]}(\theta_m(\tau)) G^{[i]} \psi^{(i)} + \sum_{\tau=0}^{k} \sum_{i=0}^{\nu} M_{\tau,i} y_v^{[i]}(\tau)$$

where $M_{\tau,i}$ are the $N \times 2^i$ sub-matrices of $M_\tau$, for all $\tau = 0, 1, \ldots, k$. Notice that the second term in the summation is deterministic and completely known at time $k$, so that the overall expression is a $\nu$-order polynomial transformation of sequence $\{y_v(\tau), \tau = 0, 1, \ldots, k\}$, which is collected into vector $Y_v^k$. Therefore, it is obvious that if an affine transformation of $\mathcal{Y}_k^\nu$, $z = \bar{m} + M\mathcal{Y}_k^\nu$, exists, then a $\nu$-order polynomial transformation of $Y_v^k$, $z = p^\nu(Y_v^k)$, also exists, and vice versa, i.e.

$$\mathcal{L}(\mathcal{Y}_k^\nu, N) \equiv P^\nu(Y_v^k, n).$$

As a consequence, KF returns $\hat{X}(k|k) = \Pi(\mathcal{X}(k)|P^\nu(\mathcal{Y}_k^\nu, N))$, whose first $n$-dimensional sub-vector is the $\nu$-PVME, $\hat{x}_{\nu-PVM}(k) = \Pi(x(k)|P^\nu(\mathcal{Y}_k^\nu, N))$, as defined in (4.5). ■

Algorithm 4.2 reports the procedure of the $\nu$-PVMKF for the 2-DTP. The computational complexity of this algorithm is $O(n^{3\nu})$, which is less negligible than that of LVMKF. A trade-off between the filter order, which may yield improvements of the estimate accuracy, and the computational burden must be established.
4.3. \( \nu \)-PVMKF for the 2-DTP

1: given: \( A, F, \psi_{\rho}^{(i)}, \psi_{\theta}^{(i)}, \rho_{m}(k), \theta_{m}(k), \forall k = 0, 1, \ldots, T, \ i = 1, 2, \ldots, 2\nu \)

2: Compute time-invariant quantities: \( A \) by (4.38), \( S^{[i]} \) and \( G \) by (4.10), \( \bar{V}_{\rho} \), \( \bar{R}^{(i)} \), and \( \bar{\Lambda} \) by Lemma 4.6, and \( \Psi_{\rho} \) and \( \Psi_{\theta} \) by Lemma 4.8

3: Initial state statistics: \( \mu_{X}(0) = \bar{X}_{0} \) and \( \Psi_{X}(0) = \Psi_{X_{0}} \)

4: Initial estimate: \( \hat{X}(0|−1) = \bar{X}_{0}, \ P(0|−1) = \Psi_{X_{0}} \)

5: for \( k = 0 \rightarrow T \) do

6: ▶ Correction

7: Compute \( \Psi_{X}(k) \) by (4.56), (4.59)-(4.61)

8: Compute \( C(k) \) by (4.39) and (4.42)

9: Compute \( \hat{y}_{v}(k) \) by (4.21)-(4.22)

10: \( K(k) = P(k|k−1)C^{T}(k) (C(k)P(k|k−1)C^{T}(k) + \Psi_{X}(k))^{-1} \)

11: \( \hat{X}(k|k) = \hat{X}(k|k−1) + K(k) \left( \hat{y}_{v}(k) − C\hat{X}(k|k−1) \right) \)

12: \( P(k|k) = (IN − K(k)C(k)) P(k|k−1) \)

13: ▶ Prediction

14: Compute \( \Psi_{W}(k) \) by (4.56), (4.57)-(4.58)

15: \( \hat{X}(k+1|k) = A\hat{X}(k|k) + U \)

16: \( P(k+1|k) = AP(k|k)A^{T} + \Psi_{W}(k) \)

17: \( \mu_{X}(k+1) = A\mu_{X}(k) + U \)

18: \( \Psi_{X}(k+1) = A\Psi_{X}(k)A^{T} + \Psi_{W}(k) + \mu_{X}(k)U^{T} + U\mu_{X}^{T}(k) \)

19: end for

Algorithm 4.2: \( \nu \)-Polynomial Virtual Measurement Kalman Filter for the 2-DTP

The quantities depending on the distribution of \( \nu_{\rho}(k) \) and \( \nu_{\theta}(k), \bar{V}_{\rho}, \bar{R}^{(i)}, \bar{\Lambda}, \Psi_{\rho}, \) and \( \Psi_{\theta}, \) are not explicitly given. However, they can be computed by using Lemmas 4.6 and 4.8 as functions of the mentioned distributions, i.e. the statistical moments \( \psi_{\rho}^{(i)} \) and \( \psi_{\theta}^{(i)}, \ i = 1, 2, \ldots, 2\nu. \)
In order to analyze the performance of the new approach, the results of a few simulations are provided in this chapter.

The considered scenario is the typical maneuvering target setting [79]. All simulations were realized by checking the behavior of the versions of the new filters LVMKF and 2-PVMKF compared to the literature filters EKF, UKF, and PF. A sample path of a Zero-Mean First-Order Markov Process [98] has been used as target trajectory (ground truth). This is characterized by two parameters indicated by $\tau$ and $\sigma_a$ (see Chapter 2). Parameter $\tau$ is the maneuver time constant that depends on how long the target maneuver lasts. Parameter $\sigma_a$ is the instantaneous standard deviation of the acceleration. For a given target trajectory, a set of 100 realizations of Gaussian measurement noises with standard deviations indicated with $\sigma_\rho$ and $\sigma_\theta$ were generated with a sampling period $\Delta = 1s$, according to the model (1.3)-(1.4). Moreover, the angle error standard deviation $\sigma_\theta$ has been considered within a set of 5 different values.

The evaluation metric of interest is the standard relative position error (RPE) which is defined for each sample measurement noise realization $i$ at time $k$ as

$$RPE_i(k) = \sqrt{\left(\left(\hat{e}_1^{(i)}(k)\right)^2 + \left(\hat{e}_2^{(i)}(k)\right)^2\right) \left((p_1(k))^2 + (p_2(k))^2\right)^{-1} \cdot 100\%},$$

where $\hat{e}_l^{(i)}(k) = p_l(k) - \hat{p}_l^{(i)}(k), l = 1, 2$, with $(\hat{p}_1^{(i)}(k), \hat{p}_2^{(i)}(k))$ being the estimated positions at time $k$. The average value of $RPE_i(k)$ with respect to the given set of samples is denoted by $RPE(k)$. The average value of $RPE(k)$ with respect to time is indicated with $RPE$. 

58
Moreover, a run is considered to be convergent only if the $RPE_i(k) < 15\%$ at the end of the simulation time.

It is well known that the filtering performances strictly depend on the accuracy of the state model. In the particular case of target tracking, this represents a crucial point since the dynamical features of all possible targets are in general unknown and hardly identifiable. The most common state models used in literature are listed in Chapter 2. For clarity, they are here summarized, assuming matrices $A$ and $F$ in (1.1) in the forms

$$A = \begin{bmatrix} \bar{A} & 0 \\ 0 & \tilde{A} \end{bmatrix}, \quad F = \begin{bmatrix} \bar{F} & 0 \\ 0 & F \end{bmatrix}. $$

1. Zero-Mean First-Order Markov model (MM) (2.6):

$$\begin{aligned}
\bar{A}_{MM} &= \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix} \frac{\tau}{\tau} \left( \Delta - \tau + \tau e^{-\Delta/\tau} \right), \\
\tilde{F}_{MM} &= \sigma_a \begin{bmatrix} \Delta^2/2 \\ 0 \end{bmatrix}.
\end{aligned} \quad (5.1)$$

$\bar{F}_{MM}$ as given in [98];

2. Wiener-Sequence acceleration model (WS) (2.5):

$$\begin{aligned}
\bar{A}_{WS} &= \begin{bmatrix} 1 & \Delta & \Delta^2/2 \\ 0 & 1 & \Delta \\ 0 & 0 & 1 \end{bmatrix}, \\
\bar{F}_{WS} &= \sigma_a \begin{bmatrix} \Delta^2/2 \\ \Delta \\ 1 \end{bmatrix}.
\end{aligned} \quad (5.2)$$

3. White-Noise Jerk model (WJ) (2.4):

$$\begin{aligned}
\bar{A}_{WJ} &= \bar{A}_{WS}, \\
\bar{F}_{WS} &= \sigma_a \sqrt{2/\tau} \begin{bmatrix} \Delta^5/20 & \Delta^4/8 & \Delta^3/6 \\ \Delta^4/8 & \Delta^3/3 & \Delta^2/2 \\ \Delta^3/6 & \Delta^2/2 & \Delta \end{bmatrix}^{1/2}.
\end{aligned} \quad (5.3)$$

4. Constant Velocity model (CV) (2.3):

$$\begin{aligned}
\bar{A}_{CV} &= \begin{bmatrix} 1 & \Delta & \Delta^2/2 \\ 0 & 1 & \Delta \\ 0 & 0 & 0 \end{bmatrix}, \\
\bar{F}_{CV} &= \begin{bmatrix} 0 \\ 0 \\ \sigma_a \end{bmatrix}.
\end{aligned} \quad (5.4)$$

Notice that model MM is the discrete-time counter-part of the model used to generate the ground truth.

For all models, $\{w(k)\}$ is assumed to be a standard white Gaussian random sequence, which means $\Psi_w = I_r$ ($r = 1$ or $r = 3$, depending on the used model). The measurement noise covariance matrix results to be equal to $\Psi_v = diag(\sigma^2_v, \sigma^2_\theta)$. These quantities are sufficient to define the model for implementing EKF, UKF and PF. For new algorithms, parameters $\bar{R}$ and $E(\bar{R}_v^{[2]}(k))$ for the LVMKF (line 2 of Algorithm 4.1) and $\bar{V}_\rho, \bar{R}_\rho^{(i)}, \bar{\Lambda}, \Psi_\rho$, and $\Psi_\theta$ for 2-PVMKF (line 2 of Algorithm 4.2) were set by applying rules in Lemmas 4.6 and 4.8, with
5.1 Simulation 1: Accuracy and Computational Time

Table 5.1 shows the numerical results in terms of RPE, together with the average computational time (ACT) per measurement noise realization, of a typical simulation whose parameters are: $\tau = 10s$, $\sigma_a = 0.32$, and $\sigma_\rho = 0.35m$, initial position $(5km, 6km)$, initial velocity $(-10m/s, -20m/s)$, zero initial acceleration, and simulation time $T_s = 100s$. Results were averaged over the set of runs resulting convergent for all the considered algorithms. Fig. 1

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>$\sigma_\theta$ (rad)</th>
<th>ACT (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.87e-2 1.31e-2 1.75e-2 2.18e-2 2.62e-2</td>
<td></td>
</tr>
<tr>
<td>LVMKF</td>
<td>0.34 0.46 0.63 0.69 0.89</td>
<td>0.21</td>
</tr>
<tr>
<td>2-PVMKF</td>
<td>0.34 0.46 0.62 0.67 0.88</td>
<td>1.97</td>
</tr>
<tr>
<td>UKF</td>
<td>0.33 0.45 0.59 0.67 0.88</td>
<td>0.04</td>
</tr>
<tr>
<td>EKF</td>
<td>0.33 0.49 0.92 1.96 4.02</td>
<td>0.02</td>
</tr>
<tr>
<td>PF</td>
<td>0.52 0.78 0.90 2.15 2.44</td>
<td>11.76</td>
</tr>
</tbody>
</table>

Table 5.1: Simulation 1: RPE and Average Computational Time (ACT).

Gaussian assumption, from which it follows that:

$$
\psi_{\rho}^{(i)} = \begin{cases} 
(i - 1)!\sigma_\rho^i & \text{if } i \text{ odd} \\
0 & \text{if } i \text{ even}
\end{cases},
$$

(Eq. 5.5)

$$
E \left[ e^{i{l}_i(q){n}_\theta(k)} \right] = e^{-\left( (i(q)\sigma_\theta)^2 \right)/2}, \quad i = 1, 2, 3, 4.
$$

(Eq. 5.6)

EKF, UKF, and PF were implemented using their suitable versions for planar tracking described in Sections 3.3.1, 3.4.3, and 3.5.4, respectively. More precisely:

- EKF was implemented without tuning;
- UKF was implemented using the modified version with $\alpha = 10^{-3}$, $\beta = 2$, and $\kappa = 0$, as recommended for the Gaussian case, and without tuning about the noises’ covariance matrices;
- PF was implemented using the SIS procedure with $N_{th} = 2/3N_p$ and systematic low-variance resampling [1]. Gaussian importance density, and number of particles $N_p = 3000$. In this case, tuning has been required in order to get acceptable results in most of the simulations. More specifically the state noise covariance has required to be over-dimensioned by setting $\eta_w = 10$.

It is worth stressing that, excepting for PF, no tuning was carried out. This choice has been made in order to get a consistent comparison.

All simulations were computed for several different trajectories (i.e. different realizations of the state noise), obtaining strictly similar results. The implementations of the filtering procedures were realized on a MATLAB© platform and computed with an Intel(R) Core(TM) i7 CPU 2.70 GHz, 8GB RAM.

5.1 Simulation 1: Accuracy and Computational Time
5.2. Simulation 2: Robustness Analysis

Figure 5.1: Simulation 1: graphical comparison of 2-PVMK with UKF (left) and PF (right) \([\sigma_\theta = 1.75e - 2\text{ rad}, \sigma_\rho = 0.35m]\).

![Graphical comparison of 2-PVMK with UKF and PF.](image)

<table>
<thead>
<tr>
<th>(\sigma_\theta (\text{rad}))</th>
<th>0.87e-2</th>
<th>1.31e-2</th>
<th>1.75e-2</th>
<th>2.18e-2</th>
<th>2.62e-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>LVMKF, 2-PVMKF</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>UKF</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>EKF</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>PF</td>
<td>13</td>
<td>27</td>
<td>46</td>
<td>49</td>
<td>76</td>
</tr>
</tbody>
</table>

Table 5.2: Simulation 1: failure percentages.

graphically represents an example of the estimation results for one of the measurement realizations. All filters were implemented by using the MM reference model.

It is evident that LVMKF, 2-PVMKF and UKF have a similar behaviour and the inadequacy of the standard EKF, especially for significant values of angle errors. It is here important to focus on the computational complexity. Recall that, for \(n\) denoting the dimension of the state vector, LVMKF is \(O(n^3)\), as well as KF and UKF, \(\nu\)-PVMKF is \(O(n^2\nu)\), and PF is \(O(N_p n^2)\). PF and \(\nu\)-PVMKF share a crucial property: on the one hand, they improve the estimation performance with a larger number of particles \(N_p\) or a larger filter order \(\nu\), respectively; on the other hand, this increases their computational burden. However, as shown in Table 5.1, the execution time of PF in this reference case \((N_p = 3000)\) is significantly higher than those of the other filters, although the RPE values are similar to those of the EKF. Moreover, Table 5.2 reports the percentage of not convergent runs for the considered algorithms. It can be noted that PF failures generally increase with the measurement error. In contrast, both UKF and the new filters have no failures.

5.2 Simulation 2: Robustness Analysis

In order to evaluate the robustness of the new filtering approach, two different checking methods have been applied. The first one consists in simulating all filtering process using each one of the state models (5.1)-(5.4). In the second one, the performances of the considered filtering
5.2. Simulation 2: Robustness Analysis

Table 5.3: Simulation 2: numerical RPE results.

<table>
<thead>
<tr>
<th>( \sigma_\theta \ (\text{rad}) )</th>
<th>0.87e-2</th>
<th>1.31e-2</th>
<th>1.75e-2</th>
<th>2.18e-2</th>
<th>2.62e-2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MM</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LVMKF</td>
<td>0.30</td>
<td>0.43</td>
<td>0.56</td>
<td>0.67</td>
<td>0.78</td>
</tr>
<tr>
<td>2-PVMKF</td>
<td>0.30</td>
<td>0.42</td>
<td>0.55</td>
<td>0.66</td>
<td>0.77</td>
</tr>
<tr>
<td>UKF</td>
<td>0.28</td>
<td>0.39</td>
<td>0.54</td>
<td>0.63</td>
<td>0.82</td>
</tr>
<tr>
<td>EKF</td>
<td>0.33</td>
<td>1.16</td>
<td>3.09</td>
<td>6.72</td>
<td>13.97</td>
</tr>
<tr>
<td><strong>WS</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LVMKF</td>
<td>0.34</td>
<td>0.49</td>
<td>0.64</td>
<td>0.78</td>
<td>0.90</td>
</tr>
<tr>
<td>2-PVMKF</td>
<td>0.33</td>
<td>0.48</td>
<td>0.63</td>
<td>0.77</td>
<td>0.90</td>
</tr>
<tr>
<td>UKF</td>
<td>0.32</td>
<td>0.49</td>
<td>0.67</td>
<td>0.81</td>
<td>1.00</td>
</tr>
<tr>
<td>EKF</td>
<td>2.77</td>
<td>11.23</td>
<td>21.42</td>
<td>33.60</td>
<td>48.60</td>
</tr>
<tr>
<td><strong>WJ</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LVMKF</td>
<td>0.33</td>
<td>0.47</td>
<td>0.61</td>
<td>0.74</td>
<td>0.87</td>
</tr>
<tr>
<td>2-PVMKF</td>
<td>0.32</td>
<td>0.47</td>
<td>0.60</td>
<td>0.73</td>
<td>0.86</td>
</tr>
<tr>
<td>UKF</td>
<td>0.33</td>
<td>0.52</td>
<td>0.81</td>
<td>1.11</td>
<td>1.45</td>
</tr>
<tr>
<td>EKF</td>
<td>1.61</td>
<td>8.60</td>
<td>17.30</td>
<td>29.70</td>
<td>45.47</td>
</tr>
<tr>
<td><strong>CV</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LVMKF</td>
<td>0.51</td>
<td>0.68</td>
<td>0.79</td>
<td>0.90</td>
<td>0.99</td>
</tr>
<tr>
<td>2-PVMKF</td>
<td>0.50</td>
<td>0.67</td>
<td>0.77</td>
<td>0.89</td>
<td>0.96</td>
</tr>
<tr>
<td>UKF</td>
<td>0.60</td>
<td>0.86</td>
<td>1.30</td>
<td>1.74</td>
<td>2.56</td>
</tr>
<tr>
<td>EKF</td>
<td>0.66</td>
<td>1.25</td>
<td>2.12</td>
<td>4.66</td>
<td>10.61</td>
</tr>
</tbody>
</table>

procedures were checked by varying the model parameters from the correct ones, using the most accurate reference model MM.

Recall that the CV is the simplest model but it is only suitable for representing constant velocity targets; WJ and WS (CA models) are more appropriate for describing maneuvering targets but not really suitable for regular constant velocity movements; finally the MM has a wider coverage than the CV and CA models. This justifies the use of the continuous time counterpart of MM for generating the ground truth and makes CV, WJ and WS less precise in describing the target behaviour. More precisely, for the used values of \( \sigma_\theta \), CV can be here considered as the crudest model, whereas WJ ans WS have to be seen as intermediate models.

Table 5.3 shows the numerical results in terms of RPE of a typical simulation whose parameters are: \( \tau = 10\text{s} \), \( \sigma_\theta = 0.32 \), and \( \sigma_\rho = 0.35\text{m} \), initial position \((12\text{km}, 8\text{km})\), initial velocity \((-10\text{m/s}, -20\text{m/s})\), zero initial acceleration, and simulation time \( T_s = 100\text{s} \). In this case, results were averaged over all the 100 runs and PF has been omitted taking into account the previous discussion. It is clear that LVMKF, 2-PVMKF and UKF have similar behaviour for low values of angular noise variance with reference to models MM, WS and WJ, while, with the increasing of the noise angle variance both LVMKF and 2-PVMKF present growing performances with respect to UKF. As far as the CV model is concerned, the performances of the new filters are definitely superior in all conditions. Also in these simulations, the EKF behaviour is decidedly worse than that of the other filters. This is confirmed by Table 5.4 where
5.3. Simulation 3: Comparison of LVMKF and 2-PVMKF

<table>
<thead>
<tr>
<th>$\sigma_\theta (\text{rad})$</th>
<th>0.87e-2</th>
<th>1.31e-2</th>
<th>1.75e-2</th>
<th>2.18e-2</th>
<th>2.62e-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>MM</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>19</td>
</tr>
<tr>
<td>WS</td>
<td>7</td>
<td>24</td>
<td>49</td>
<td>67</td>
<td>82</td>
</tr>
<tr>
<td>WJ</td>
<td>3</td>
<td>24</td>
<td>42</td>
<td>67</td>
<td>76</td>
</tr>
<tr>
<td>CV</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>12</td>
</tr>
</tbody>
</table>

Table 5.4: Simulation 2: EKF failure percentage.

<table>
<thead>
<tr>
<th>$\sigma_\theta (\text{rad})$</th>
<th>0.87e-2</th>
<th>1.31e-2</th>
<th>1.75e-2</th>
<th>2.18e-2</th>
<th>2.62e-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>LVMKF</td>
<td>0.35</td>
<td>0.48</td>
<td>0.62</td>
<td>0.74</td>
<td>0.85</td>
</tr>
<tr>
<td>2-PVMKF</td>
<td>0.35</td>
<td>0.48</td>
<td>0.62</td>
<td>0.74</td>
<td>0.84</td>
</tr>
<tr>
<td>UKF</td>
<td>0.37</td>
<td>0.61</td>
<td>1.00</td>
<td>1.30</td>
<td>1.74</td>
</tr>
<tr>
<td>EKF</td>
<td>0.59</td>
<td>2.01</td>
<td>5.18</td>
<td>11.10</td>
<td>22.48</td>
</tr>
</tbody>
</table>

Table 5.5: Simulation 2: robustness analysis.

the percentage of not convergent runs are presented, tacking into account that both UKF and the new filters have no failures.

In the second checking method, parameter $\sigma_a$ has been assumed as a random variable endowed with a log-uniform distribution, having the real standard deviation $\sigma_a^*$ as mean value. The above described simulations were repeated on the same data by setting parameter $\sigma_a$ at each of the values in the set \{\$2\sigma_a^* 2i/2 : i = −8, −7, \ldots, 7, 8\}. Finally, the robustness was computed as the average of the RPEs so obtained. Table 5.5 reports such an index referred to the scenario of simulation 1 showing the high quality in terms of state estimation and robustness of the proposed algorithms.

5.3 Simulation 3: Comparison of LVMKF and 2-PVMKF

In all the reported simulation results, 2-PVMKF has better estimation performances with respect to the LVMKF, as it is expected from the theoretical argumentation. However, in most cases, they exhibit very similar RPE values. In order to stress the effective difference between the two algorithms, a nongaussian setting was considered. In particular, the above described scenario was reproduced by generating uniform distributed measurement noises instead of the conventional Gaussian ones. Recalling that $\sigma_\rho$ and $\sigma_\theta$ denote the standard deviations of such distributions, the radius and angle error sequences were uniformly generated in the intervals $[-\sqrt{3}\sigma_\rho, \sqrt{3}\sigma_\rho]$ and $[-\sqrt{3}\sigma_\theta, \sqrt{3}\sigma_\theta]$, respectively. Moreover, parameters in (5.5) and (5.6) were substituted by

$$
\psi_\rho^{(1)} = \psi_\rho^{(3)} = 0, \quad \psi_\rho^{(2)} = \sigma_\rho^2, \quad \psi_\rho^{(4)} = \frac{9}{5}\sigma_\rho^4,
$$

$$
E\left[e^{j\lambda_i(q)n_\theta(k)}\right] = \frac{\sin(l_i(q)\sqrt{3}\sigma_\theta)}{l_i(q)\sqrt{3}\sigma_\theta}, \quad i = 1, 2, 3, 4.
$$
Table 5.6: Simulation 3: numerical RPE results for nongaussian noise.

<table>
<thead>
<tr>
<th>$\sigma_\theta$ (rad)</th>
<th>0.87e-2</th>
<th>1.31e-2</th>
<th>1.75e-2</th>
<th>2.18e-2</th>
<th>2.62e-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>MM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LVMKF</td>
<td>0.39</td>
<td>0.57</td>
<td>0.72</td>
<td>0.81</td>
<td>0.94</td>
</tr>
<tr>
<td>2-PVMKF</td>
<td>0.35</td>
<td>0.51</td>
<td>0.65</td>
<td>0.75</td>
<td>0.88</td>
</tr>
<tr>
<td>WS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LVMKF</td>
<td>0.37</td>
<td>0.54</td>
<td>0.64</td>
<td>0.76</td>
<td>0.86</td>
</tr>
<tr>
<td>2-PVMKF</td>
<td>0.35</td>
<td>0.49</td>
<td>0.59</td>
<td>0.72</td>
<td>0.83</td>
</tr>
<tr>
<td>WJ</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LVMKF</td>
<td>0.36</td>
<td>0.51</td>
<td>0.63</td>
<td>0.72</td>
<td>0.83</td>
</tr>
<tr>
<td>2-PVMKF</td>
<td>0.33</td>
<td>0.47</td>
<td>0.57</td>
<td>0.67</td>
<td>0.79</td>
</tr>
<tr>
<td>CV</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LVMKF</td>
<td>1.85</td>
<td>2.32</td>
<td>2.70</td>
<td>2.86</td>
<td>3.14</td>
</tr>
<tr>
<td>2-PVMKF</td>
<td>1.60</td>
<td>2.06</td>
<td>2.46</td>
<td>2.68</td>
<td>2.99</td>
</tr>
</tbody>
</table>

Table 5.6 reports the numerical results in terms of RPE of a simulation whose parameters are: $\tau = 10s$, $\sigma_a = 0.32$, and $\sigma_\rho = 3.5m$, initial position $(4km, 4km)$, initial velocity $(-5m/s, -10m/s)$, zero initial acceleration, and simulation time $T_s = 100s$. In this case the only new algorithms have been considered. Since no failure occurred for both algorithms, RPEs were averaged over all the 100 runs. As one can note, in this nongaussian setting the polynomial algorithm clearly overperforms the linear one, especially when filtering uses less accurate state models such as WJ and CV. Fig. 5.2 graphically represents an example of the estimation results for one of the measurement realizations where the improved performance of the 2-PVMKF algorithm with respect to the LVMKF can be observed.
In this thesis, the 2-D tracking problem has been considered exploiting the idea of “virtual measurement map”. This consists in transforming the nonlinear measurement output function into a linear time-varying map corrupted by an additive non-Gaussian noise. The resulting model is amenable to the application of Kalman filter as linear least square estimator with respect to the defined virtual output map, without any linearization. Such a filter has been named Linear Virtual Measurement Kalman Filter (LVMKF). Moreover, in order to manage the non-Gaussian noise, a polynomial extension has been designed, which could improve the performance of the linear filter. This second algorithm has been named \( \nu \)-Polynomial Virtual Measurement Kalman Filter (\( \nu \)-PVMKF), where \( \nu \) is the filter order.

The effectiveness of the proposed solutions has been checked by numerical simulations which reproduced several typical tracking settings. Three popular State of the Art tracking algorithms (EKF, UKF and PF) have been used for comparison. Results have been discussed in terms of estimation accuracy, robustness with respect to model errors, and computational burden, as summarized in the following.

- **Estimation accuracy**: The new filters clearly outperform the EKF and avoid bad behaviours such as divergence, especially for high noise levels. Whereas, for standard settings, the performance of LVMKF, \( \nu \)-PVMKF, and UKF are comparable.

- **Robustness**: Model errors have been introduced for simulating real-world scenarios and study the capability of the filtering techniques to preserve accuracy. The new filters show a good robustness since they both preserve the performance level when models are...
wrong. In the same conditions, EKF has bad performances and often diverges, whereas UKF has acceptable results, but the estimation accuracy gets worse when errors increase.

- **Computational burden**: LVMKF is very fast, as well as EKF and UKF, since they all have rather the same computational complexity. Although 𝜈-PVMKF has a less negligible computational cost, the 2-PVMKF completes the filtering procedure with a time sufficiently low to be processed in a real-framework. On the contrary, PF requires a significantly larger time to obtain acceptable results.

Based on this discussion, it can be concluded that the proposed approach is able to avoid the drawbacks caused by linearization and also is robust with respect to possible model errors, with a computational time comparable to literature algorithms.

Future works may regard the experimental validation and the extension to three dimensional tracking.
Throughout this thesis, Kronecker algebra is widely used [9, 18, 19]. Here, for the sake of completeness, some definitions and properties are recalled and some new results on this subject are given.

**Definition A.1.** Let $M$ and $N$ be real matrices of dimensions $r \times s$ and $p \times q$ respectively. Then, the Kronecker product $M \otimes N$ is defined as the $(r \cdot p) \times (s \cdot q)$ matrix

$$M \otimes N = \begin{bmatrix}
m_{1,1}N & \cdots & m_{1,s}N \\
\vdots & \ddots & \vdots \\
m_{r,1}N & \cdots & m_{r,s}N
\end{bmatrix},$$

where $m_{i,j}$ are the entries of $M$.

Of course, this kind of product is not commutative.

**Definition A.2.** Let $M$ be a $r \times s$ real matrix, the Kronecker power of $M$ is defined as

$$M^{[0]} = 1 \in \mathbb{R},$$

$$M^{[l]} = M \otimes M^{[l-1]} \quad l \geq 1.$$ 

**Definition A.3.** Let $M$ be the $r \times s$ real matrix

$$M = \begin{bmatrix}m_1 & m_2 & \cdots & m_s\end{bmatrix},$$
where \( m_i \) denotes the \( i \)-th column of \( M \). Then, the stack of \( M \) is the \( r \cdot s \) vector 

\[
\text{st}(M) = \begin{bmatrix} m_1^T & m_2^T & \cdots & m_s^T \end{bmatrix}^T.
\]

Moreover, the inverse operation is defined as 

\[
\text{st}^{-1} \left( \begin{bmatrix} m_1^T & m_2^T & \cdots & m_s^T \end{bmatrix}^T \right) = M.
\]

It can be easily proved that the stack operator is linear. Some useful properties of the Kronecker product and power are given in the following.

**Lemma A.4.** For any matrices \( A, B, C, \) and \( D \), and any vectors \( u \) and \( v \), it is:

\[
\begin{align*}
(A + B) \otimes (C + D) &= (A \otimes C) + (A \otimes D) + (B \otimes C) + (B \otimes D), \\
(A \otimes B) \cdot (C \otimes D) &= (A \cdot C) \otimes (B \cdot D), \\
(A \otimes B)^T &= A^T \otimes B^T, \\
\text{st}(A \cdot B \cdot C) &= (C^T \otimes A) \cdot \text{st}(B), \\
\text{st}(uv^T) &= v \otimes u,
\end{align*}
\]

the \( \cdot \) denoting the standard matrix product.

For the proof see [9].

**Lemma A.5.** For any given pair of matrices \( A \in \mathbb{R}^{r \times s}, B \in \mathbb{R}^{n \times m} \) it results:

\[
A \otimes B = C_{r,n}^T (A \otimes B) C_{s,m},
\]

where the \( (h, l) \) entry of the commutation matrix \( C_{u,v} \in \mathbb{R}^{(uv) \times (uv)} \) is given by

\[
(C_{u,v})_{h,l} = \begin{cases} 1, & \text{if } l = (|h - 1|_{\nu}) u + (\left\lfloor \frac{h-1}{\nu} + 1 \right\rfloor); \\ 0, & \text{otherwise}. \end{cases}
\]

For the proof see [19].

**Lemma A.6.** Let \( x, y \in \mathbb{R}^n \), for any integer \( h \), it is:

\[
(x + y)^[h] = \sum_{k=0}^{h} M_{h,n}^k (x^{[k]} \otimes y^{[h-k]}),
\]

where \( M_{h,n}^k \) satisfies

\[
\begin{align*}
M_{h,n}^h &= M_{0,n}^h = I_n^{[h]}, \\
M_{k,n}^h &= (M_{k-1,n}^{h-1} \otimes I_n) + (M_{k-1,n}^{h-1} \otimes I_n)(I_n^{[k-1]} \otimes G_{h-k}), \quad 1 \leq k \leq h,
\end{align*}
\]

with \( G_l \) given by

\[
\begin{align*}
G_1 &= C_{n,n}^T, \\
G_l &= (I_n \otimes G_{l-1})(G_1 \otimes I_{l-1}), \quad l > 1,
\end{align*}
\]
where $C_{n,n}^T$ is given by (A.7).

For the proof see [19].

**Lemma A.7.** Let $a_i \in \mathbb{R}^n$, $1 \leq i \leq p$, $p \in \mathbb{N}$. Then:

$$(a_1 + a_2 + \ldots + a_p)^[h] = \sum_{h_1, \ldots, h_p \geq 0 \atop h_1 + \cdots + h_p = h} M_{h_1, \ldots, h_p}^h (a_1^{[h_1]} \otimes a_2^{[h_2]} \otimes \cdots \otimes a_p^{[h_p]}),$$

where $M_{h_1, \ldots, h_p}^h \in \mathbb{R}^{n \times n^h}$ satisfies

- $M_{h_1, \ldots, h_p}^h = 0_{n^h}$ if at least one of $h_i$ is negative,
- $M_{h_1, \ldots, h_p}^h = I_n$ for $h = 1$,
- $M_{h_1, \ldots, h_p}^h = \sum_{1 \leq i \leq p-1} \left( M_{h_1, \ldots, h_i-1, \ldots, h_p}^h \otimes I_n \right) \left( I_n^{[h_1, \ldots, h_i-1]} \otimes G_{h_{i+1}, \ldots, h_p} \right) + \left( M_{h_1, \ldots, h_{p-1}}^h \otimes I_n \right)$ for $h > 1$.

with $G_i$ given by (A.8).

For the proof see [19].

**Lemma A.8.** Let $Z \in \mathbb{R}^r$, $r = n(n^\nu - 1)/(n - 1)$, $\nu, n \in \mathbb{N}$, be defined as in (4.24). Then:

$$Z^{[h]} = \sum_{h_1, \ldots, h_{\nu} \geq 0 \atop h_1 + \cdots + h_{\nu} = h} M_{h_1, \ldots, h_{\nu}}^h \left[ \left( \bigotimes_{i=1}^{\nu} (\Pi_{\nu,n}^{(i)} z[i])^{[h_i]} \right) z^{[h_1 + 2h_2 + \cdots + \nu h_{\nu}]} \right].$$

**Proof.** Vector $V$ can be represented as

$$Z = \sum_{i=1}^{\nu} \Pi_{\nu,n}^{(i)} z[i].$$

Therefore, from Lemma A.7 and iteratively using property (A.2) it follows that

$$Z^{[h]} = \left( \sum_{i=1}^{\nu} \Pi_{\nu,n}^{(i)} z[i] \right)^{[h]} = \sum_{h_1, \ldots, h_{\nu} \geq 0 \atop h_1 + \cdots + h_{\nu} = h} M_{h_1, \ldots, h_{\nu}}^h \left[ \bigotimes_{i=1}^{\nu} (\Pi_{\nu,n}^{(i)} z[i])^{[h_i]} \right]$$

$$= \sum_{h_1, \ldots, h_{\nu} \geq 0 \atop h_1 + \cdots + h_{\nu} = h} M_{h_1, \ldots, h_{\nu}}^h \left[ \left( \bigotimes_{i=1}^{\nu} (\Pi_{\nu,n}^{(i)} z[i])^{[h_i]} \right) z^{[h_1 + 2h_2 + \cdots + \nu h_{\nu}]} \right].$$

**Lemma A.9.** Let $z \in \mathbb{R}^n$. Then, for any $i \geq 0$, the $q$-th entry if $z[i]$ is

$$(z[i])_q = z_{l_1} z_{l_2} \cdots z_{l_q},$$
where
\[ l_m = \left\lfloor \frac{q - 1}{n^{m-1}} \right\rfloor + 1, \quad m = 1, 2, \ldots, i - 1, \]
\[ l_q = |q - 1|_n + 1. \]  
(A.9)

For the proof see [18].

**Lemma A.10.** Let \( x, y \in \mathbb{R}^n \), \( A, B \in \mathbb{R}^{n \times n} \), and
\[
\mathcal{X} = \begin{bmatrix} x^{[1]} \\ \vdots \\ x^{[\nu]} \end{bmatrix}, \quad \mathcal{Y} = \begin{bmatrix} y^{[1]} \\ \vdots \\ y^{[\nu]} \end{bmatrix}, \quad [Ax]_e = \begin{bmatrix} A^{[1]}x^{[1]} \\ \vdots \\ A^{[\nu]}x^{[\nu]} \end{bmatrix}.
\]

For any \( h = 1, 2, \ldots, \nu, \nu \geq 1 \), it is:
\[ [Ax]_e = \Lambda_{\nu}(A)\mathcal{X}, \]  
\[ (x + y)^{[h]} = \mathcal{M}_{\nu,n}^{(h)}(\mathcal{X} \otimes \mathcal{Y}) + x^{[h]} + y^{[h]}, \]  
\[ (Ax + By)^{[h]} = \mathcal{M}_{\nu,n}^{(h)}(\Lambda_{\nu}(A) \otimes \Lambda_{\nu}(B))(\mathcal{X} \otimes \mathcal{Y}) + (Ax)^{[h]} + (By)^{[h]}, \]  
where matrices \( \mathcal{M}_{\nu,n}^{(h)} \in \mathbb{R}^{n^h \times r^2} \), \( r = n(\nu - 1)/(n - 1) \) are such that
\[ \mathcal{M}_{\nu,n}^{(1)} = 0, \]
\[ \mathcal{M}_{\nu,n}^{(h)} = \sum_{k=1}^{h-1} M_{k,n}^h \left( \Pi_{\nu,n}^{(k)} \otimes \Pi_{\nu,n}^{(h-k)} \right), \quad h > 1. \]  
(A.13)

**Proof.** Equation (A.10) is trivially satisfied. For the case \( h = 1 \), (A.11) holds true. For \( h > 1 \), by consecutively applying Lemma A.6, property (4.27) and property (A.2), it results:
\[
(x + y)^{[h]} = \sum_{k=1}^{h-1} M_{k,n}^h \left( \left( \Pi_{\nu,n}^{(k)} \mathcal{X} \otimes \Pi_{\nu,n}^{(h-k)} \mathcal{Y} \right) + x^{[h]} + y^{[h]} \\
= \sum_{k=1}^{h-1} M_{k,n}^h \left( \Pi_{\nu,n}^{(k)} \otimes \Pi_{\nu,n}^{(h-k)} \right) (\mathcal{X} \otimes \mathcal{Y}) + x^{[h]} + y^{[h]} \\
= \mathcal{M}_{\nu,n}^{(h)} (\mathcal{X} \otimes \mathcal{Y}) + x^{[h]} + y^{[h]},
\]
which proves (A.11). By applying last property to vectors \( Ax \) and \( By \), one has:
\[
(x + y)^{[h]} = \mathcal{M}_{\nu,n}^{(h)} ([Ax]_e \otimes [By]_e) + (Ax)^{[h]} + (By)^{[h]},
\]
that can be rewritten exploiting properties (A.10) and (A.2) as
\[
(x + y)^{[h]} = \mathcal{M}_{\nu,n}^{(h)} ((\Lambda_{\nu}(A)\mathcal{X}) \otimes (\Lambda_{\nu}(B)\mathcal{Y})) + (Ax)^{[h]} + (By)^{[h]} \\
= \mathcal{M}_{\nu,n}^{(h)} ((\Lambda_{\nu}(A) \otimes \Lambda_{\nu}(B))(\mathcal{X} \otimes \mathcal{Y}) + (Ax)^{[h]} + (By)^{[h]},
\]
which corresponds to (A.12). ■
List of Figures

1.1 2-DTP: graphical representation. ................................................. 3

2.1 Ternary-uniform mixture distribution. ................................. 13

3.1 The mean and standard deviation ellipses for the actual and linearized form of the transformation from polar to Cartesian coordinates (Example 3.1). The true mean is at × and the uncertainty ellipse is solid. Linearization calculates the mean at ○ and the uncertainty ellipse is dotted. (Figure courtesy of [66]). 25

3.2 The principle of unscented transform. ........................................... 26

3.3 The mean and standard deviation ellipses for the actual, linearized, and unscented form of the transformation from polar to Cartesian coordinate (Example 3.1). The true mean is at × and the uncertainty ellipse is dotted. The position of unscented mean is indicated by + and its ellipse is solid. Linearization calculates the mean at ○ and the uncertainty ellipse is dotted. (Figure courtesy of [66]). 29

3.4 The mean and standard deviation ellipses for the actual and MC approximation forms of the transformation from polar to Cartesian coordinates (Example 3.1). The true mean is at × and the uncertainty ellipse is solid. The positions of the mean values of the MC approximations with \( N_p = 3.5 \times 10^2 \), \( 3.5 \times 10^3 \), and \( 3.5 \times 10^4 \) are indicated by ○, □, and +, respectively. The ellipses are dotted, point-dashed, and dashed, respectively. ............................................. 34

3.5 Illustration of how \( u(j) \) are chosen with multinominal, stratified, and systematic resampling, in this case with 10 particles (Figure Courtesy of [56]). ........ 39
4.1 The polynomial filtering approach: the optimal estimate is the orthogonal projection of the state \( x \) onto the space \( L^2 \) (dotted line). The suboptimal estimates are: the projection \( \hat{x}_L \) onto the space of linear transformations of the output \( L \) (dot-dashed); the projection \( \hat{x}_2 \) onto the space of 2nd order polynomial transformations of the output \( P^2 \) (short-dashed); the projection \( \hat{x}_3 \) onto the space of 3rd order polynomial transformations of the output \( P^3 \) (long-dashed). As the polynomial order increases the distance (i.e. the error variance) becomes smaller. ................................................................. 44

5.1 Simulation 1: graphical comparison of 2-PVMK with UKF (left) and PF (right) 
\[ \sigma_\theta = 1.75e - 2rad, \ \sigma_\rho = 0.35m \] ................................................................. 61

5.2 Simulation 3: graphical comparison of LVMKF with 2-PVMK \[ \sigma_\theta = 2.18e - 2rad, \ \sigma_\rho = 3.5m \] ................................................................. 64
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>Simulation 1: RPE and Average Computational Time (ACT).</td>
<td>60</td>
</tr>
<tr>
<td>5.2</td>
<td>Simulation 1: failure percentages.</td>
<td>61</td>
</tr>
<tr>
<td>5.3</td>
<td>Simulation 2: numerical RPE results.</td>
<td>62</td>
</tr>
<tr>
<td>5.4</td>
<td>Simulation 2: EKF failure percentage.</td>
<td>63</td>
</tr>
<tr>
<td>5.5</td>
<td>Simulation 2: robustness analysis.</td>
<td>63</td>
</tr>
<tr>
<td>5.6</td>
<td>Simulation 3: numerical RPE results for nongaussian noise.</td>
<td>64</td>
</tr>
</tbody>
</table>


