Czech Technical University in Prague
Faculty of Nuclear Sciences and Physical Engineering

Knowledge transfer for linear systems with bounded noise

Přenos znalosti pro lineární systémy s omezeným šumem

Master's Thesis

Author: **Bc. Eva Lainová**

Supervisor: **Ing. Lenka Kuklišová Pavelková, Ph.D.**

Academic year: 2021/2022

Katedra: matematiky Akademický rok: 2020/2021

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student:	Bc. Eva Lainová
Studijní program:	Aplikace přírodních věd
Studijní obor:	Aplikované matematicko-stochastické metody
Název práce (česky):	Přenos znalosti pro lineární systémy s omezeným šumem
Název práce (anglicky):	Knowledge transfer for linear systems with bounded noise

Pokyny pro vypracování:

1) Navažte na váš výzkumný úkol „Přenos znalosti mezi bayesovskými filtry“.
2) Nastudujte a popište techniku přenosu znalosti využívající plně pravděpodobnostní návrh (PPN) aplikovanou na stavový model s omezeným šumem.
3) Nastudujte techniky fúze informace (FI) pro stavové modely s omezeným šumem.
4) Porovnejte metodu PPN se dvěma vybranými metodami FI, a sice jednou centralizovanou a jednou distribuovanou.
5) Příslušné algoritmy sestavte v jazyce Matlab a testujte na datech pro lokalizaci objektu ve 2-D prostoru.
6) Diskutujte výhody a nevýhody jednotlivých metod.

Doporučená literatura:

1) T. Meng, X. Jing, Z. Yan, W. Pedrycz, A survey on machine learning for data fusion. Information fusion 57, 2020, 115-129.
2) L. Jirsa, L. Kuklišová Pavelková, A. Quinn, Bayesian transfer learning between uniformly modelled Bayesian filters. In 'Informatics in Control, Automation and Robotics ', Springer International Publishing, Cham, 2021, 151-168.
3) B. Chen, D. W. C. Ho, W. Zhang, L. Yu, Networked fusion estimation with bounded noises. IEEE Transactions on Automatic Control 62, 2017, 5415-5421.
4) Q. Shen, J. Liu, X. Zhou, W. Qin, L. Wang, Q. Wang, Centralized Fusion Methods for Multi-Sensor System With Bounded Disturbances. IEEE Access 7, 2019, 141612-141626.

Jméno a pracoviště vedoucího diplomové práce:

Ing. Lenka Kuklišová Pavelková, Ph.D.
ÚTIA AV ČR, v.v.i., Pod Vodárenskou věží 4, 182 08 Praha 8 ,

Jméno a pracoviště konzultanta:

Datum zadání diplomové práce: 28.2.2021

Datum odevzdání diplomové práce: 5.1.2022

Doba platnosti zadání je dva roky od data zadání.

V Praze dne 28.2.2021

[illegible]
garant oboru

[illegible]
vedoucí katedry

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta jaderná a fyzikálně inženýrská
-3-

[illegible]
děkan

Acknowledgment:
I would like to thank my thesis advisor Ing. Lenka Kuklišová Pavelková, Ph.D. for her expert guidance.

Author's declaration:
I declare that this Master's Thesis is entirely my own work and I have listed all the used sources in the bibliography.

Prague, May 2, 2022

Bc. Eva Lainová

[illegible]

Název práce:

Přenos znalosti pro lineární systémy s omezeným šumem

Autor: Bc. Eva Lainová

Obor: Aplikované matematicko-stochastické metody

Druh práce: Diplomová práce

Vedoucí práce: Ing. Lenka Kuklišová Pavelková, Ph.D., ÚTIA AV ČR, v.v.i., Pod Vodárenskou věží 4, 182 08 Praha 8

Abstrakt: Tato práce popisuje a porovnává tři metody pro odhadování stavů, které využívají informace ze dvou nezávislých senzorů. Cílem všech tří metod je penalizovat méně přesné informace a zvýšit přesnost odhadu stavu lineárního systému s omezeným šumem. Výsledky odhadů jsou porovnány se simulovanými daty. Na dvou nezávislých senzorech jsou zaznamenána reálna data o poloze, rychlosti a zrychlení pohybujícího se objektu. Naměřená data jsou zpracována a všechny tři metody jsou otestovány na reálných datech. Na závěr jsou uvedeny výsledky experimentů pro simulovaná i reálná data.

Klíčová slova: Bayesovský odhad stavů, centralizovaná fúze znalostí, distribuovaná fúze znalostí, Kullback-Leiblerova divergence, lineární stavový model, systém dvou senzorů, omezený šum, plně pravděpodobnostní návrh, přenos znalosti, rovnoměrné rozdělení pravděpodobnosti

Title:

Knowledge transfer for linear systems with bounded noise

Author: Bc. Eva Lainová

Abstract: This Master's Thesis describes and compares three methods for state estimation which use information from two independent sensors. Linear system with bounded noise is considered. All methods aim to penalize less precise information to prevent negative transfer and increase estimation precision. Experiments are conducted on simulated data. Real data on position, speed and acceleration of a moving object are collected and processed. All estimation methods are applied on processed real data. The results of all experiments are presented.

Key words: Bayesian state estimation, bounded noise, centralised information fusion, distributed information fusion, fully probabilistic design, knowledge transfer, Kullback-Leibler divergence, linear state-space model, two-sensor system, uniform probability distribution

Contents

Introduction

Machine learning and data mining techniques have achieved a great success in many applications, including pattern recognition, image processing, speech recognition and others. Traditional machine learning methods perform best under the assumption that training data and test data are from the same feature space and same distribution. In cases, this assumption is not met, the results are degraded and most models need to be rebuilt from scratch. In real world, it is sometimes impossible or too expensive to collect sufficient data or to rebuilt the models [18]. In order to reduce the need of recollecting new training data it would be useful to transfer knowledge across domains. This is exactly what knowledge transfer does.

Knowledge transfer is based on using knowledge from previous tasks to improve learning on a new task. The idea for knowledge transfer between related tasks was inspired by human inborn ability to learn and to solve completely new tasks using previous experience. For example consider two people one of whom is a guitar player the other has no experience with musical instruments. Which one is more likely to learn to play the violin faster? Of course, it is the first one. Thanks to his experience with the guitar he can transfer his music knowledge onto the new task which is playing the violin [29]. The domains from which we draw data must be related for transfer learning to be applicable. For example the ability to drive a car probably will not help when a person starts to learn how to play the violin. Operating a car and playing music are unrelated tasks therefore knowledge of one will not improve ability to learn the other.

Information fusion is a technique combining data from multiple sources to improve accuracy of single source measurements. There are two types of information fusion: centralized [27] and distributed [7] information fusion. Both fusion methods aim to provide an optimal estimate. Centralized information fusion uses raw data from all sensors which are sent to the fusion centre to provide global estimate. Distributed information fusion firstly processes data from each sensor locally to obtain local estimate and subsequently fuses the local estimates in fusion centre. In the presence of faulty sensors, centralised fusion has poorer robustness and reliability comparing to distributed fusion [24].

Information fusion has many real-world applications e.g. target tracking, object localization, navigation, sensor networks, image processing, etc. [17]. In real-world application, information fusion sometimes faces the problems of data sparsity (missing data) in the target domain. There are insufficient data in the domain of interest but there are enough data in another domain. These data may be from different feature space and/or follow different distribution. The solution to this problem is to use knowledge transfer which was designed to deal exactly with this difficulty. To transfer knowledge from source domain to target domain transfer learning-based information fusion is used [30].

Generally, state estimation research concentrates on systems with white Gaussian noise. In linear system, the most commonly use method is application of Kalman filter. In real-world engineering practice the noise statistical properties are usually unknown or non-Gaussian. In these cases, Kalman filter is no longer applicable [11].

In this thesis, the knowledge transfer will be applied on state estimation of linear systems with bounded noise. This thesis follows up on the author's Bachelor thesis [16] where state estimation of constrained linear system was studied. The state estimation on an isolated filter in [16] was based on fully probabilistic design (FPD). This work looks into state estimation with knowledge transfer. The whole framework is extended from one filter to two filters where the first filter (target filter) obtains knowledge from the second filter (source filter). The target optimizes the decision making task based on the received knowledge by minimizing Kulblack-Leibler divergence. This approach to optimisation problem is known as fully probabilistic design (FPD).

Aim of the thesis is to analyse and compare three state estimation techniques for linear systems with bounded noise. The first estimation method is based on knowledge transfer via fully probabilistic design. The other two are based on centralised and distributed information fusion. Furthermore, the contribution of knowledge transfer in state estimation in [16] is evaluated. Theoretical analysis as well as experiments are provided. Experiments are conducted on simulated data as well as on real data.

This thesis is organised as follows. In the first chapter, the preliminary information (notation used, definitions etc.) for better orientation in the thesis are given. The following chapter is dedicated to Bayesian knowledge transfer based on fully probabilistic design. Distributed measurement fusion method is described in the third chapter. In the fourth chapter, centralised measurement fusion method for state estimation is described. Experiments setups and results of all methods are found in the fifth chapter. Finally, the results of experiments on simulated and real data are compared and discussed.

Chapter 1

Preliminaries

1.1 Notation

In this paper following notation is used. Capital letters A are appointed to matrices. Vectors and scalars are in lower case b. A_{ij} is the element of matrix A in the i-th row and j-th column. The symbol $f(\cdot \mid \cdot)$ denotes a conditional probability density function (pdf). $\propto$ means equality up to a constant factor. $\mathrm{tr}(\cdot)$ is the matrix trace operator. $\otimes$ symbolizes the Kronecker product. $\oplus$ stands for Minkowski sum. $\mathbb{A}$ stand for set of a, $a \in \mathbb{A}$. $\mathcal{V}$ denotes volume. $\hat{x}$ symbolizes the estimate of vector x. $\mathcal{U}(\cdot,\cdot)$ stands for probability distribution with appropriate parameters. $\dot{z}$ stands for derivative of z. The number of dots represent to order of the derivative i.e., $\ddot{z}$ is the second order derivative of z and $\dddot{z}$ is the third order derivative of z.

1.2 Linear state space model

Consider state space model in following form

$$\begin{aligned} x(t) &= Ax(t-1) + Bu(t-1) + v(t), \\ y_j(t) &= C_j x(t) + w_j(t), \end{aligned} \tag{1.1}$$

where $j = 1, 2$. t represents the discrete time index, $t \in \mathbb{T}$, $\mathbb{T} = \{1, \dots, \bar{t}\}$, $\bar{t}$ is the final time step. $x(t) \in \mathbb{R}^{l_x}$ stands for the state vector. $y_j(t) \in \mathbb{R}^{l_{y_j}}$ is the output vector of the j-th sensor also known as the observation vector. $u(t) \in \mathbb{R}^{l_u}$ is the known control input vector. A, B and C_j are parameter matrices of suitable dimensions and C_j is the parameter matrix of the measurement equation of the j-th sensor. $v(t)$ represents state noise and $w_j(t)$ output noise of the j-th sensor. $v(t)$ and $w_j(t)$ are assumed to be independent identically distributed (i.i.d.) [20], [7]. The first equation in (1.3) is called the state equation, the second is called the measurement equation.

The following section is dedicated to multiple geometric representation of support of the uniform distribution.

1.3 Geometric representations of support

The support of a function $f(z)$ is defined as set $\mathbb{Z}_f\{z : f(z) \neq 0\}$.

The geometric definitions of examples of supports of the uniform distribution which will be used in further in this thesis are drawn from [12].

A convex polytope is convex set which is formed by finite number of flat hyperplanes.

A zonotope $\mathbb{Z}_Z$ is a special case of convex polytope, where upper and lower bounds are given by vectors a, b of length k and V is a matrix of size $k \times l_z$ and rank l_z. A zonotope is in following form

$$\mathbb{Z}_Z = \{z : a \leq Vz \leq b\}. \tag{1.2}$$

Let zonotope be defined as an intersection of $k \geq l_z$ strips $\mathbb{Z}_S$.

A strip $\mathbb{Z}_S$ is defined as

$$\mathbb{Z}_{S_i} = \{z : a_i \leq Vz_i \leq b_i\}, \tag{1.3}$$

where $i = 1, \ldots, l_z$ and a_i, b_i is the i-th component of vector a, b, respectively.

A parallelotope $\mathbb{Z}_P$ is a special case of zonotope. Parallelotope is also an intersection of strips (1.3) where $k = l_z$ i.e. a, b are vectors of length l_z, V is an nonsingular square matrix of size $l_z \times l_z$. Parallelotope can be expressed in numerous forms. The first is identical to (1.2)

$$\mathbb{Z}_P = \{z : a \leq Vz \leq b\}. \tag{1.4}$$

The second is following

$$\mathbb{Z}_P = \{z : -\mathbf{1}_{l_z} \leq Wz - c \leq \mathbf{1}_{l_z}\}, \tag{1.5}$$

where $\mathbf{1}_{l_z}$ is unit vector of length l_z. W and c are defined as

$$W_{ij} = \frac{2V_{ij}}{b_i - a_i}, \qquad c_i = \frac{b_i + a_i}{b_i - a_i}. \tag{1.6}$$

where $i = 1, \ldots, l_z$.

The next form of a parallelotope is

$$\mathbb{Z}_P = \{z : z = z_c + T\xi\}, \tag{1.7}$$

where $\forall \xi$ s.t. $||\xi||_\infty = \max_i(\xi_i) < 1$. This form is called the centroid form.

Let the centroid z_c be defined as

$$z_c = Tc, \tag{1.8}$$

considering $a = c - \mathbf{1}_{l_z}$, $b = c + \mathbf{1}_{l_z}$ and $V = W$, $T = W^{-1}$.

The volume of a paralletope is computed as follows

$$\mathcal{V}_P = \mid \det V \mid \prod_{i=1}^{l_z} (b_i - a_i). \tag{1.9}$$

An orthotope $\mathbb{Z}_O$ is a specific case of a parallelotope Z_P and therefore also a specific case of a zonotope $\mathbb{Z}_Z$ in (1.5) where $V = I$, I denotes an identity matrix. a and b are lower and upper bounds

$$\mathbb{Z}_O = \{z : a \leq z \leq b\}. \tag{1.10}$$

Orthotope can be equivalently expressed in forms (1.5) and (1.7) using $V = I$.

The volume of an orthotope $\mathbb{Z}_O$ is computed as follows

$$\mathcal{V}_O = \prod_{i=1}^{l_z} (b_i - a_i). \tag{1.11}$$

An ellipsoid $\mathbb{Z}_E$ is a set given by the form

$$\mathbb{Z}_E = \{z : (z-d)^T E(z-d) \leq 1\}, \tag{1.12}$$

where E is a positive definite matrix od size $l_z \times l_z$ and d is vector of size l_z. Vector d is the centre of the ellipsoid. Volume of ellipsoid $\mathbb{Z}_E$ given by E as in (1.12) has following form

$$\mathcal{V}_E = \mathcal{V}_{l_z}(\det(E))^{-\frac{1}{2}}, \tag{1.13}$$

where $\mathcal{V}_{l_z}$ is the unity ball in $\mathbb{R}^{l_z}$ [26].

1.4 Uniform distribution

In this section, a multidimensional uniform distribution is defined.

$\mathcal{U}$ symbolizes uniform probability density function (pdf) of a given variable. To define uniform distribution and analyse its support characteristic function is introduced.

Characteristic function $\chi(z)$. Consider set $\mathbb{Z} = \{z : a \leq Vz \leq b\}$, where $a, b \in \mathbb{R}^k$, V is a $k \times n$, $k \geq n$ matrix of rank n, $z \in \mathbb{R}^n$ and $Vz : \mathbb{R}^n \to \mathbb{R}^k$ is a continuous linear mapping. Let the characteristic function $\chi(z)$ on set $\mathbb{Z}$ be defined as

$$\chi(z) = \begin{cases} 1 & \text{for} \quad z \in \mathbb{Z}, \\ 0 & \text{otherwise} \end{cases} \tag{1.14}$$

Equivalent notation of the characteristic function (1.14) is

$$\chi(z) \equiv \chi(a \leq Vz \leq b). \tag{1.15}$$

The set $\mathbb{Z}$ is the support of characteristic function (1.14). It directly depends on the dimension of V. Let the matrix V be of size $k \times n$, $k \geq n$ and of rank n.

Let $\mathcal{U}_z(a \leq Vz \leq b)$ be a uniform pdf of a random variable z with parallelotopic support (1.4).

$$\mathcal{U}_z(a \leq Vz \leq b) = K\chi(a \leq Vz \leq b), \tag{1.16}$$

where K is normalizing constant and $\chi(a \leq Vz \leq b)$ is characteristic function in form (1.15). The volume $\mathcal{V}$ of the set $\mathbb{Z}$ is finite. Since $\mathcal{V}$ is the volume of set $\mathbb{Z}$, the normalizing constant from (1.16) $K = \mathcal{V}^{-1}$. This implies that set $\mathbb{Z}$ is a convex set bounded by a finite number of hyperplanes.

Using following notation the normalising constant can be omitted

$$\mathcal{U}_z(a \leq Vz \leq b) \propto \chi(a \leq Vz \leq b).$$

In this thesis appears following equivalent notation pdf of uniform distribution

$$\mathcal{U}_z(a \leq Vz \leq b) \equiv \mathcal{U}_z(a, b, V).$$

In case $V = I$, where I is the identity matrix, the pdf support is an orthotope (1.10). The notation is following

$$\mathcal{U}_z(a \leq z \leq b) \equiv \mathcal{U}_z(a, b). \tag{1.17}$$

If the pdf support is an ellipsoid (1.12) then the notation is

$$\mathcal{U}_z((z-d)^T E(z-d) \leq 1) \propto \chi((z-d)^T E(z-d) \leq 1).$$

1.5 Fully probabilistic design

This section is a brief introduction to fully probabilistic design which will be applied on knowledge transfer problem in Chapter 2.

Fully probabilistic design (FPD) is an alternative to control design for stochastic systems which is traditionally based on optimization of the expected value of a suitably chosen loss function. The FPD approach proposed in [14] leads to simpler equations with lower computational complexity. The control design is based on minimising Kullback-Leibler distance also known as Kullback-Leibler divergence (KLD). Consider p and q to be probability densities. The Kullback-Leibler distance (KLD) between p and q is defined as follows

$$KLD(p \parallel q) = \mathrm{E}_p \left[\ln \left(\frac{p}{q} \right) \right], \tag{1.18}$$

where E_p is the expected value with respect to p.

FPD aims to find the unknown distribution F from knowledge constrained distribution set $\mathbb{F}$. The ideal distribution f_I is a zero measure which means that $KLD(f_I \parallel f_I) = 0$. Optimal distribution f_O in FPD sense is the $F \in \mathbb{F}$ which has the smallest KLD

$$\mathrm{KLD}(F \parallel f_I) = \mathrm{E}_F \left[\ln \left(\frac{F}{f_I} \right) \right], \tag{1.19}$$

to the ideal distribution f_I. To find f_O following minimization task needs to be solved

$$f_O = \arg\min_{F \in \mathbb{F}} \mathrm{KLD}(F \parallel f_I). \tag{1.20}$$

The optimal distribution f_O is the one with minimal KLD to the ideal distribution f_I.

In [15], the FPD is further elaborated and formally justified. Axiomatic mathematical background for FPD is developed. It shows that FPD is a proper extension of the standard Bayesian decision making. FPD unifies and simplifies subtasks of decision making under uncertainty.

In [14] and [15], KLD between actual joint pdf and an ideal one is minimised. Although this theoretical approach was primarily developed for control design it found its application in knowledge transfer. The FPD can be used to find the optimal pdf for knowledge transfer between two filters. The FPD approach is applied to knowledge transfer in papers [9], [19], [23], [12] .

Chapter 2

Fully probabilistic design for knowledge transfer

In the considered Bayesian set up [22], the system is given by following pdfs

$$\begin{aligned} \text{prior pdf:} \quad & f(x(1)) \\ \text{observation model:} \quad & f(y(t) \mid x(t)) \\ \text{time evolution model:} \quad & f(x(t+1) \mid x(t), u(t)) \end{aligned} \tag{2.1}$$

where $x(t)$ is an unobservable l_x-dimensional system state, $y(t)$ is a l_y-dimensional observable output, $u(t)$ is a l_u-dimensional known control input and t represents time step, $t \in \mathbb{T}$.

It is assumed that system states x_t satisfy Markov property and no direct relationship exists between system inputs and outputs in the observation model in (2.1).

2.1 Bayesian state estimation

Bayesian state estimation or filtering is the evolution of the posterior pdf $f(x(t) \mid \mathbf{d(t)})$ where $\mathbf{d(t)} = (d(1), \ldots, d(t))$ is a sequence of observed data records $d(t) = \{y(t), u(t)\}$ from starting time $t = 1$ until t, $t \in \mathbb{T}$ i.e., all data (input and output) acquired up until time t. The evolution of $f(x(t) \mid \mathbf{d(t)})$ has two components. It is described by a two-step recursion (time update and data update) which starts from the prior pdf $f(x(1))$ and ends by data update at the final time $\bar{t}$.

Data update:

$$f(x(t) \mid \mathbf{d(t)}) = \frac{f(y(t) \mid x(t)) f(x(t) \mid \mathbf{d(t\text{-}1)})}{\int_{\mathbb{X}(t)} f(y(t) \mid x(t)) f(x(t) \mid \mathbf{d(t\text{-}1)}) \mathrm{d}x(t)} = \frac{f(y(t) \mid x(t)) f(x(t) \mid \mathbf{d(t\text{-}1)})}{f(y(t), x(t) \mid \mathbf{d(t\text{-}1)})} \tag{2.2}$$

Time update:

$$f(x(t+1) \mid \mathbf{d(t)}) = \int_{\mathbb{X}(t)} f(x(t+1) \mid u(t), x(t)) f(x(t) \mid \mathbf{d(t)}) \mathrm{d}x(t), \tag{2.3}$$

where $\mathbb{X}(t)$ is the support of function $f(x(t) \mid \mathbf{d(t)})$.

The general description of stochastic system in (2.1) is given a specific form (1.1). The state noise and output noise $v(t)$ and $w_j(t)$ are both assumed to be mutually independent and uniformly distributed on finite support.

$$\begin{aligned} f(v(t)) &= \mathcal{U}_v(-\nu, \nu), \\ f(w_j(t)) &= \mathcal{U}_{w_j}(-\omega_j, \omega_j), \end{aligned} \tag{2.4}$$

where $\omega_j \in \mathbb{R}^{l_{y_j}}$ and $\nu \in \mathbb{R}^{l_x}$ are vectors of positive components and form the support bounds. Symbol $\mathcal{U}$ represents uniform pdf on an orthotopic support described in (1.17).

Considering the uniform distribution on noises (2.4) the state space model has following pdf form equivalent to (1.1)

$$\begin{aligned} f(x(t) \mid x(t-1)) &= \mathcal{U}_x(Ax(t-1) + Bu(t-1) - \nu, Ax(t-1) + Bu(t-1) + \nu), \\ f(y_j(t) \mid x(t)) &= \mathcal{U}_{y_j}(C_j x(t) - \omega_j, C_j x(t) + \omega_j). \end{aligned} \tag{2.5}$$

Approximated data update and time update is described for single isolated filter hence the notation of the measurement equation (1.1) is simplified to y, C and ω.

Approximated data update: Data update (2.2) processes the observation model $f(y(t) \mid x(t))$ (2.1) together with the pdf $f(x(t) \mid \mathbf{d(t\text{-}1)})$ resulting from previous time update (2.3) and current observation $y(t)$. The recursion starts at the time $t = 1$ with the prior pdf $f(x(1))$ (2.1) which is uniform on an orthotopic support i.e.,

$$f(x(1)) = \mathcal{U}_{x(1)}(\underline{x}(1), \overline{x}(1)). \tag{2.6}$$

The result of the data update according to [22] is a posterior pdf $f(x_t \mid d(t))$ with polytopic support. The newly arisen polytope is a result of an intersection of an orthotope given by the previous time update $f(x(t) \mid \mathbf{d(t\text{-}1)})$ or in the first time step by prior pdf $f(x(1))$ and l_y strips (1.3) given by the new observation $y(t)$

$$\begin{aligned} f(x(t) \mid d(t)) =& H\chi(\underline{m}(t) - \nu \leq x(t) \leq \overline{m}(t) + \nu) \times \chi(Cx(t) - \omega \leq y(t) \leq Cx(t) + \omega) = \\ =& H\chi(\underline{m}(t) - \nu \leq x(t) \leq \overline{m}(t) + \nu) \times \chi(y(t) - \omega \leq Cx(t) \leq y(t) + \omega) \end{aligned} \tag{2.7}$$

where H is a normalizing constant and can be omitted using symbol $\propto$ then

$$\begin{aligned} f(x(t) \mid d(t)) \propto& \chi(\underline{m}(t) - \nu \leq x(t) \leq \overline{m}(t) + \nu) \times \chi(y(t) - \omega \leq Cx(t) \leq y(t) + \omega) \\ =& \chi\left(\begin{bmatrix} \underline{m}(t) - \nu \\ y(t) - \omega \end{bmatrix} \leq \begin{bmatrix} I \\ C \end{bmatrix} x(t) \leq \begin{bmatrix} \overline{m}(t) + \nu \\ y(t) + \omega \end{bmatrix}\right). \end{aligned} \tag{2.8}$$

I symbolizes an identity matrix of size $l_x \times l_x$, $\underline{m}(t)$ and $\overline{m}(t)$ are obtained as

$$\begin{aligned} \underline{m}_i(t) &= \sum_{k=1}^{l_x} \min(A_{ik}\underline{x}_k(t-1) + B_{ik}u_k(t-1), A_{ik}\overline{x}_k(t-1) + B_{ik}u_k(t-1)), \\ \overline{m}_i(t) &= \sum_{k=1}^{l_x} \max(A_{ik}\underline{x}_k(t-1) + B_{ik}u_k(t-1), A_{ik}\overline{x}_k(t-1) + B_{ik}u_k(t-1)), \end{aligned} \tag{2.9}$$

where $\underline{m}_i(t)$, $\overline{m}_i(t)$ is the i-th component of the vector $\underline{m}(t)$, $\overline{m}(t)$, respectively. $\underline{x}_k(t-1)$, $u_k(t-1)$ is the k-th component of the vector $\underline{x}(t-1)$, $u(t-1)$, respectively.

As was already mentioned, the resulting pdf $f(x(t) \mid \mathbf{d(t)})$ from (2.8) has polytopic support. To keep the pdf support in intended form i.e. orthotopic the following approximation process introduced in [21] is used.

As an intermediate step between polytopic and orthotopic form the polytope is circumscribed by a parallelotope. The algorithm is briefly described in [21] and with detailed theoretical background found in [28].

A short description is provided. The second row in (2.8) is actually a pdf with orthotopic support since it corresponds with the second row in (2.5) which is uniform pdf with orthotpic support by definition. The second term in (2.8) is also an orthotope (explanation will be provided in the following part: Approximated time update). Therefore their intersection is generally a polytope. Since orthotope is a special case of a parallelotope, the support of the pdf on the second row of (2.5) can be considered a parallelotope.

The orthotope forming the support of the second term in (2.8) can be represented by l_y data strips (1.3) in $x(t)$ space

$$\mathbb{Z}_{S_i} = \{x(t) : y_i(t) - \omega_i \leq C_i x(t) \leq y_i(t) + \omega_i\}, \tag{2.10}$$

where $i = 1 \dots l_y$. $y_i(t)$, ω_i is the i-th component of the vector $y(t)$, ω, respectively. C_i is the i-th row of the matrix C.

The parallelotope in the second term of (2.8) can be also represented by l_x stripes in $x(t)$ space

$$\mathbb{Z}_{S_i} = \{x(t) : \underline{m}_i(t) - \nu_i \leq x_i(t) \leq \overline{m}_i(t) + \nu_i\}, \tag{2.11}$$

where $i = 1 \dots l_x$. $\underline{m}_i(t)$, ν_i, $x_i(t)$, $\overline{m}_i(t)$ is the i-th component of the vector $\underline{m}(t)$, ν, $x(t)$, $\overline{m}(t)$, respectively.

The general intersection of an orthotope and parallelotope in (2.8) is a polytope but the approximation of this intersection, according to [28], is done strip by strip. Firstly, to the paralelotope given by l_x strips in (2.11) is added one of the l_y strips from (2.10). The $l_x + 1$ strips are narrowed and/or shifted to remove redundancies but so that their intersection remains unchanged. Then one of the $l_x + 1$ strips is removed so that the intersection of the l_x strips that remain has the minimal volume. The volume of a parallelotope is computed in (1.15). The process of adding and removing strips one by one is repeated for all l_y. This way it is ensured that the form remains parallelotopic and therefore the result is support in form of a parallelotope. The resulting paralelotop is in form

$$f(x(t) \mid \mathbf{d(t)}) \approx K(t)\chi(\underline{x}^*(t) \leq M(t)x(t) \leq \bar{x}^*(t)), \tag{2.12}$$

where $\underline{x}^*(t)$ and $\bar{x}^*(t)$ are the lower and upper bounds of the parallelotope and M is the nonsingular identity matrix as defined in (1.4).

The aim is to provide an orthotopic approximation hence another circumscription is necessary. To obtain the required bounds $\underline{x}(t)$ and $\overline{x}(t)$ of the orthotope

$$\underline{x}(t) \leq x(t) \leq \overline{x}(t) \tag{2.13}$$

the following computation process from [21] is performed. Firstly, note that the paralletope ($\underline{x}^*(t) \leq M(t)x(t) \leq \bar{x}^*(t)$) given in (2.12) has an equivalent notation $\{x : -\mathbf{1}_{(l_x)} \leq W(t)x(t) - c(t) \leq \mathbf{1}_{(l_x)}\}$ according to (1.5) where $\mathbf{1}_{(l_x)}$ is the unit vector of length l_x. $W(t)$ and $c(t)$ are computed according to (1.6) as follows

$$W_{ij}(t) = \frac{2M_{ij}(t)}{\bar{x}^*_i(t) - \underline{x}^*_i(t)}, \qquad c_i(t) = \frac{\bar{x}^*_i(t) + \underline{x}^*_i(t)}{\bar{x}^*_i(t) - \underline{x}^*_i(t)}, \tag{2.14}$$

where $\underline{x}^*_i(t)$, $\bar{x}^*_i(t)$ is the i-th component of the vector $\underline{x}^*(t)$, $\bar{x}^*(t)$, respectively.

Define $T(t)$ as $T(t) = W(t)^{-1}$ and the centroid (1.8) $x_c(t)$ as $x_c(t) = T(t)c(t)$. Now, the parallelotope can be expressed in the centroid form (1.7) $x(t) = x_c(t) + T(t)\xi$ where $\forall \xi$ s.t. $||\xi||_\infty = \max_i(\xi_i) < 1$. The sum of absolute values of the $i-$th row in matrix $T(t)$ gives the $i-$th coordinate of the vertex most distant from the center point in $i-$th direction. The sum represents the half of the width of the orthotope in $i-$th direction that contains the parallelotope. To find all coordinates of the orthotope that tightly circumscribes the parallelotope, absolute values in all rows in $T(t)$ must be summarized

$$Q_{ii}(t) = \sum_{j=1}^{l_x} |T_{ij}(t)|. \tag{2.15}$$

The resulting $Q(t)$ is a diagonal matrix, where the diagonal elements are the sum th absolute values of corresponding rows in the $T(t)$ matrix.

The orthotope resulting from the previous computation is defined in the centroid form as $x(t) = x_c(t) + Q(t)\xi$ which is equivalent to

$$-\mathbf{1}_{l_x} \leq Q^{-1}(t)x(t) - x_c(t) \leq \mathbf{1}_{l_x}. \tag{2.16}$$

The pdf bounds (2.13) are

$$\underline{x}(t) = -Q(t)\mathbf{1}_{l_x} + x_c(t), \quad \overline{x}(t) = Q(t)\mathbf{1}_{l_x} + x_c(t). \tag{2.17}$$

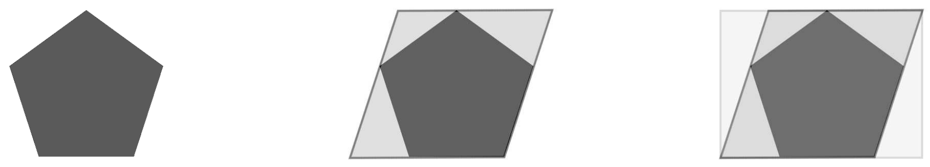
The approximated pdf has the form

$$f(x(t) \mid \mathbf{d(t)}) \approx \mathcal{U}_x(\underline{x}(t), \overline{x}(t)). \tag{2.18}$$

The state point estimate $\hat{x}_t$ is the centre of the orthotope

$$\hat{x}(t) = \frac{\underline{x}(t) + \overline{x}(t)}{2}. \tag{2.19}$$

In the two-dimensional task, the approximation of the support of the pdf during data update demonstrates the following picture sequence 2.1.



(a) Polytope in (2.8) (b) Parallelotope in (2.12) (c) Orthotope in (2.18)

Figure 2.1: Approximation sequence of the support of pdf after data update for two-dimensional task

Approximated time update: During the time update (2.3) the time evolution model $f(x(t+1) \mid x(t), u(t))$ from (2.1) and the result of previous data update $f(x(t) \mid \mathbf{d(t)})$ are processed. The time update exactly computed in [21]. The resulting pdf has an orthotopic support but is not uniformly distributed. To keep the pdf in the given class of uniform distribution on an orthotopic support an approximation is in introduced [21]. The original trapezoidal pdf is approximated by a uniform distribution by minimising the Kullback-Leibler divergence of two pdfs [21]. The resulting approximation has the form

$$f(x(t) \mid \mathbf{d(t\text{-}1)}) \approx \prod_{i=1}^{l_x} \frac{\chi(\underline{m}_i(t) - \nu_i \leq x_i(t) \leq \overline{m}_i(t) + \nu_i)}{\overline{m}_i(t) - \underline{m}_i(t) + 2\nu_i}, \tag{2.20}$$

where $\underline{m}_i(t)$, $\overline{m}_i(t)$ computed in (2.9) are the i-th components of vector $\underline{m}(t)$, $\overline{m}(t)$ which are of the same length as the state vector $x(t)$ i.e., $x(t), \underline{m}(t), \overline{m}(t) \in \mathbb{R}^{l_x}$ The result of the approximation in [21] is the pdf $f(x(t) \mid \mathbf{d(t\text{-}1)})$ with a linear piecewise shape

$$f(x(t) \mid \mathbf{d(t\text{-}1)}) \approx \mathcal{U}_x(\underline{m}(t) - \nu, \overline{m}(t) + \nu). \tag{2.21}$$

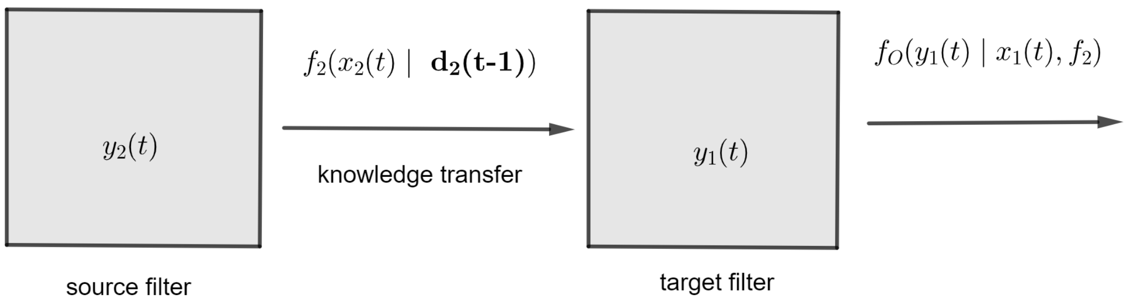

$y_2(t)$
source filter
$f_2(x_2(t) \mid \mathbf{d_2(t\text{-}1)})$
knowledge transfer
$y_1(t)$
target filter
$f_O(y_1(t) \mid x_1(t), f_2)$

Figure 2.2: Knowledge transfer schema

2.2 Knowledge transfer

If not stated otherwise, this section draws on [23]. Consider two stochastically independent filters, the source filter and the target filter. Each filter models its local environment. Further, to distinguish between the two filters, the state and data belonging to the target filter will be labelled x_1, $\mathbf{d_1}$ and the state and data belonging to the source filter will be labelled x_2, $\mathbf{d_2}$. To improve the performance of the target filter $\{x_1, \mathbf{d_1}\}$, the probabilistic knowledge about the source filter $\{x_2, \mathbf{d_2}\}$ environment is transferred into the target filter. In other words, to the isolated filter from the previous section, a source filter is added with the aim to improve the state estimation.

Each filter models its local system given by its states x_1 and x_2, outputs and inputs $\mathbf{d_1}$ and $\mathbf{d_2}$, respectively. As depicted in the Figure 2.2, the target filter has access to the information about the source filter environment only in form of the state predictor f_2 not to the data $\mathbf{d_2}$ itself. In contrast to the isolated filter from previous section, where the filter has information regarding only one filter, the target filter in two sensor-system has information in form of data $\mathbf{d_1}$ and in form of the state predictor of the source filter f_2. This is the main difference between the isolated filter and the pair-filter system. The knowledge transfer between the pair of filters is performed via FPD.

To compare isolated filtration and two-filter estimation, recall the isolated filter problem. The total information about state and output evolution that holds the isolated filter can be expressed as joint probability distribution function

$$f(y(t), x(t) \mid \mathbf{d(t\text{-}1)}) = f(y(t) \mid x(t)) f(x(t) \mid \mathbf{d(t)}) \tag{2.22}$$

the numerator in (2.2).

In the target-source filter task, the knowledge transfer is performed see Figure 2.2. The transferred information is in form of the state predictor of the source filter f_2 therefore the joint probability must be conditioned by the state predictor of the source filter $f_2 = f_2(x_2(t) \mid \mathbf{d_2(t\text{-}1)})$. The target filter joint pdf after knowledge transfer is in form

$$F(y_1(t), x_1(t) \mid \mathbf{d_1(t\text{-}1)}, f_2). \tag{2.23}$$

The use of capital F for probability distribution function should stress the fact that the joint pdf of the target filter after knowledge transfer is unknown and non-unique. The reason is that the source and target filter are independent. No model describing their relationship is considered. Then the target pdf F can be separated into two factors as in (2.22)

$$F = F(y_1(t), x_1(t) \mid \mathbf{d_1(t\text{-}1)}, f_2) = F(y_1(t) \mid x_1(t), \mathbf{d_1(t\text{-}1)}, f_2)F(x_1(t) \mid \mathbf{d_1(t\text{-}1)}, f_2), \tag{2.24}$$

where $F \in \mathbb{F}$ and $\mathbb{F}$ is a set of all possible pdfs. After separating the target pdf F into to factors the factors can be analysed individually.

Firstly, the second function on the right side of (2.24) will be analysed $F(x_1(t) \mid \mathbf{d_1(t\text{-}1)}, f_2)$. The pdf includes the knowledge that the target filter has about its state $x_1(t)$ given data $\mathbf{d_1(t\text{-}1)}$ and the state predictor of the source filter f_2 after knowledge transfer from source filter. Here, the model with full acceptance is chosen which means that the target takes the source state predictor as its own state predictor. Hence the unknown pdf $F(x_1(t) \mid \mathbf{d_1(t\text{-}1)}, f_2)$ is equivalent to the state predictor pdf of the source filter

$$F(x_1(t) \mid \mathbf{d_1(t\text{-}1)}, f_2) = f_2(x_2(t) \mid \mathbf{d_2(t\text{-}1)}). \tag{2.25}$$

Assuming the source state $x_2(t)$ and target state $x_1(t)$ are equal in distribution, then

$$f_2(x_2(t) \mid \mathbf{d_2(t\text{-}1)}) = f_2(x_1(t) \mid \mathbf{d_2(t\text{-}1)}). \tag{2.26}$$

It follows that the second factor in (2.24) is not variational but fixed.

As consequence the only factor from the right side of (2.24) which remains variational and unknown is $F(y_1(t) \mid x_1(t), \mathbf{d_1(t\text{-}1)}, f_2)$. $F(y_1(t) \mid x_1(t), \mathbf{d_1(t\text{-}1)}, f_2)$ represents the observation model (2.2) which is conditionally independent of the data sequence $\mathbf{d_1(t\text{-}1)}$. The knowledge transfer does not interfere with the conditional independence therefore

$$F(y_1(t) \mid x_1(t), \mathbf{d_1(t\text{-}1)}, f_2) = F(y_1(t) \mid x_1(t), f_2). \tag{2.27}$$

Next, let (2.27) and (2.26) be inserted back to (2.24)

$$F = F(y_1(t), x_1(t) \mid \mathbf{d_1(t\text{-}1)}, f_2) = F(y_1(t) \mid x_1(t), f_2)f_2(x_1(t) \mid \mathbf{d_2(t\text{-}1)}), \tag{2.28}$$

where factor $f_2(x_1(t) \mid \mathbf{d_2(t\text{-}1)})$ is fixed and factor $F(y_1(t) \mid x_1(t), f_2)$ is variational. $\mathbb{F}$ is the set of all possible models, therefore

$$\begin{aligned} F \in \mathbb{F} = \{&\text{set of models } F(y_1(t), x_1(t) \mid \mathbf{d_1(t\text{-}1)}, f_2) \\ &\text{with } F(y_1(t), x_1(t) \mid \mathbf{d_1(t\text{-}1)}, f_2) \text{ variational and } f_2(x_1(t) \mid \mathbf{d_2(t\text{-}1)}) \text{ fixed}\}. \end{aligned} \tag{2.29}$$

Since the set of all admissible pdfs $\mathbb{F}$ is defined, it remains to be decided which pdf from $\mathbb{F}$ should be chosen. To find the optimal pdf $f_O(y_1(t) \mid x_1(t), f_2) \in \mathbb{F}$, fully probabilistic design is applied. Before the FPD approach is used, the ideal pdf needs to be defined. For the ideal the joint pdf of the isolated filter is chosen (2.22). In the knowledge transfer task, the ideal pdf is the joint pdf of the target filter before knowledge transfer i.e.

$$f_I = f(y_1(t) \mid x_1(t))f(x_1(t) \mid \mathbf{d_1(t\text{-}1)}). \tag{2.30}$$

The variational pdf which needs to be optimized is in form

$$F = F(y_1(t) \mid x_1(t), f_2) f_2(x_1(t) \mid \mathbf{d_2(t\text{-}1)}). \tag{2.31}$$

To solve the optimization problem, FPD approach is chosen. According to FPD, the optimal pdf f_O is the one wich minimizes the Kullback-Leibler distance from F to the ideal pdf f_I which is fixed. The Kullback-Leibler distance (KLD) from F to f_I is defined in (1.19). The FPD process of finding the optimal pdf f_O includes the known constrains $f_O \in \mathbf{F}$ (2.29) and the preferences about f_O given by the ideal pdf f_I (2.30). f_O is the closest probability density to the fixed ideal density f_I in the sense of minimum Kullback-Leibler divergence (KLD) (1.20).

The knowledge $f_2(x_2(t) \mid \mathbf{d_2(t\text{-}1)})$ transferred from the source to the target can have various forms. Here, it is assumed that $f_2(x_2(t) \mid \mathbf{d_2(t\text{-}1)})$ is the state predictor resulting from time update at time $t-1$ performed simultaneously on the source filter. In contrast to the previous work on the FPD-optimal knowledge transfer [13], [23] where the transferred knowledge from source to target was the source data predictor $f_2(y_2(t) \mid \mathbf{d_2(t\text{-}1)})$ in [23] choose the authors the source state predictor $f_2(x_2(t) \mid \mathbf{d_2(t\text{-}1)})$ as the transferred knowledge. As depicted in the picture 2.2 the optimal pdf after knowledge transfer f_O does not directly include the data from the source filter $\mathbf{d_2}$. The state predictor of the source $f_2(x_2(t) \mid \mathbf{d_2(t\text{-}1)})$ includes the source data and the f_0 includes the state predictor of the source filter f_2. Since the FPD-optimal pdf is now defined, the notation can be simplified so that $f_O(y_1(t) \mid x_1(t), f_2) = f(y_1(t) \mid x_1(t), f_2)$.

The aim of this these is to address the problem of knowledge transfer between pair of Bayesian filters under bounded state and observational noise. The following part is concerned with the problem of the bounded noise.

Knowledge transfer between Bayesian filters with bounded noise

The state estimation on an isolated filter under bounded noise was described in section (2.1). The noise was assumed to be uniformly distributed with fixed bounds and an orthotopic support. The aim is to extend the filtration problem to a knowledge transfer pair-filtration with uniformly distributed noise an orthotopic support.

Even though there is an explicit minimizer which solves (2.30), the goal is to find the optimal solution of f_O for uniformly distributed with bounded support. The support is a function of $x_1(t)$.

The target's state predictor before knowledge transfer $f_1(x_1(t) \mid \mathbf{d_1(t\text{-}1)})$ is uniform on bounded support $\mathbb{X}_1(t)$. The state predictor of the source filter $f_2(x_2(t) \mid \mathbf{d_2(t\text{-}1)})$ which is to be transferred is also uniform with bounded support $\mathbb{X}_2(t)$. Then, the bounded intersection $\mathbb{X}(t)$ is defined as follows

$$\mathbb{X}(t) = \mathbb{X}_1(t) \cap \mathbb{X}_2(t). \tag{2.32}$$

The set $\mathbb{X}_1(t)$ represents the knowledge about state $x_1(t)$ before knowledge transfer and the set $\mathbb{X}_2(t)$ represents the knowledge about state $x_2(t)$ before knowledge transfer. $\mathbb{X}_1(t)$ resp. $\mathbb{X}_2(t)$ defines the part of the state-space in which the state $x_1(t)$ resp. $x_2(t)$ can be found according to information the predictor has been given. The intersection of $\mathbb{X}_1(t)$ and $\mathbb{X}_2(t)$ is the knowledge transfer. The intersection can have two results.

Case 1: $\mathbb{X}(t) \neq \emptyset$ then $\mathbb{X}(t)$ is the FPD-optimal set for $x_1(t)$ after knowledge transfer hence the optimal solution to minimizing problem (1.20) is

$$f_1(y_1(t) \mid x_1(t), f_2) = f_1(y_1(t) \mid x_1(t) \in \mathbb{X}(t)). \tag{2.33}$$

Case 2: $\mathbb{X}(t) = \emptyset$ i.e., the intersection of $\mathbb{X}_1(t)$ and $\mathbb{X}_2(t)$ is the empty set hence the knowledge from source filter has no effect on the resulting pdf i.e., the source brings no knowledge to the target. Then

$$f_1(y_1(t) \mid x_1(t), f_2) = f_1(y_1(t) \mid x_1(t)). \tag{2.34}$$

The resulting pdf is independent of f_2 which means that the resulting pdf is the same as in isolated filter task. The optimal pdf is $f_1(y_1(t) \mid x_1(t) \in \mathbb{X}_1(t))$. Proof is found in [23].

The knowledge transfer between two filters with bounded support of their predictors can be extended to knowledge transfer among n filters where the target is assumed f_1. Then for case 1 the result is the same as in (2.34). In case 2 where the intersection is non empty set, the result is following

$$f_1(y_1(t) \mid x_1(t), f_2, \ldots, f_n) = f_1(y_1(t) \mid x_1(t) \in \mathbb{X}(t)), \tag{2.35}$$

where

$$\mathbb{X}(t) = \bigcap_{i=1}^{n} \mathbb{X}_i(t). \tag{2.36}$$

For proof see [23].

Now, return to knowledge transfer between two filters. The condition laid down on the state predictor's support was boundedness, regardless of the support's shape. Let apply the condition to bounded orthotopic support.

Assume the state predictors $f_1(x_1(t) \mid \mathbf{d_1(t\text{-}1)})$ and $f_2(x_2(t) \mid \mathbf{d_2(t\text{-}1)})$ to have uniformly bounded supports with orthotopic shape (1.10). The state predictors are result of previous time update (2.21) hence the supports are orthotopic. The orthotopic sets are closed under the intersection operator (1.20). The orthotopic supports of state predictors have the form $\mathbb{X}_1(t) = \{x_1(t) : \underline{x}_1(t) \leq x_1(t) \leq \overline{x}_1(t)\}$ and $\mathbb{X}_2(t) = \{x_2(t) : \underline{x}_2(t) \leq x_2(t) \leq \overline{x}_2(t)\}$. The orthotopic sets are closed under the intersection operator (1.20) (excluding the case $\mathbb{X}(t) = \emptyset$). Then the FPD-optimal orthotopic set of $x_1(t)$ after knowledge transfer is the intersection of $\mathbb{X}_1(t)$ and $\mathbb{X}_2(t)$

$$\mathbb{X}(t) = \mathbb{X}_1(t) \cap \mathbb{X}_2(t) = \{x_1(t) : \max(\underline{x}_1(t), \underline{x}_2(t)) \leq x_1(t) \leq \min(\overline{x}_1(t), \overline{x}_2(t))\}. \tag{2.37}$$

The orthotopic sets are closed under the intersection operator because the result of every intersection is either an orthotope or the empty set. The orthotopic set is defined in (1.10). The orthotope is given only by upper and lower bounds hence the edges of the orthotope are parallel to the axes.

In two-dimensional case the result of the intersection (2.37) has four possible results as depicts the Figure 2.3. The first Figure 2.3a demonstrates the case when the source support is smaller than the target support and the bigger orthotope contains the other hence the resulting orthotope $\mathbb{X}(t) = \mathbb{X}_2(t)$ and the transfer is positive because the resulting support smaller than the targets support. The smaller the support the more accurate it the resulting estimate. This is called the positive knowledge transfer.

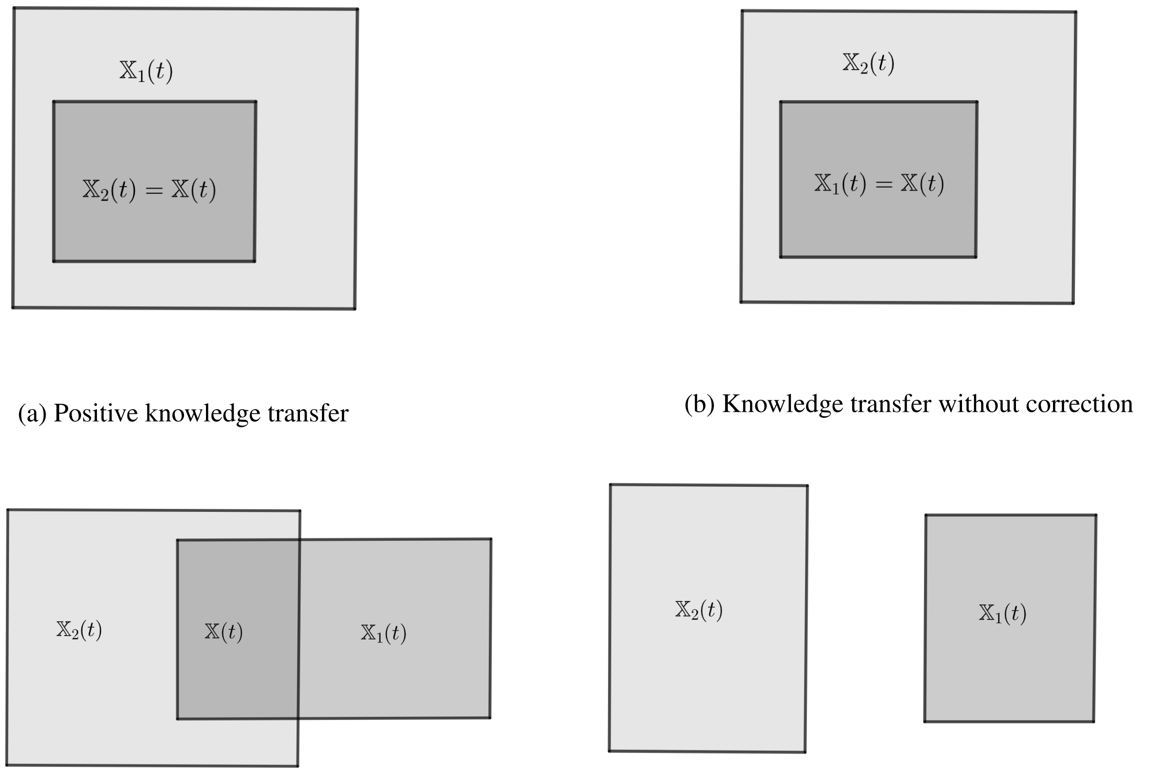
The second Figure 2.3b shows the reverse case when the source support is bigger than the target support and the bigger orthotope contains the other. The result is then $\mathbb{X}(t) = \mathbb{X}_1(t)$. In this case the knowledge transfer took place but the knowledge from the source brought no improvement to the target.

The third Figure 2.3c is similar to the first case. The resulting intersection is smaller then the target support itself hence the estimate accuracy is improved. This is also positive knowledge transfer.

The last Figure 2.3d shows the case when the intersection between $\mathbb{X}_1(t)$ and $\mathbb{X}_2(t)$ is the empty set. This case is dealt with in (2.34). If the intersection is the empty set the result is the target support hence no knowledge transfer takes place and the target filter behaves like an isolated filter.

According to (2.2) the processing of local datum $d_1(t)$ on target after knowledge transfer is in form

$$\begin{aligned} f_1(x_1(t) \mid \mathbf{d_1(t)}, f_2) &\propto f_1(y_1(t) \mid x_1(t), f_2) f_1(x_1(t) \mid \mathbf{d_1(t\text{-}1)}) \propto \\ &\propto f_1(y_1(t) \mid x_1(t)) f_1(x_1(t) \mid \mathbf{d_1(t\text{-}1)}, f_2), \end{aligned} \tag{2.38}$$



(a) Positive knowledge transfer

(b) Knowledge transfer without correction

(c) Positive knowledge transfer

(d) Intersection between $\mathbb{X}_1(t)$ and $\mathbb{X}_2(t)$ is the empty set i.e. no knowledge is transferred (2.34)

Figure 2.3: Intersection of two orthotopes in two-dimensional case.

where $f_1(x_1(t) \mid \mathbf{d_1(t\text{-}1)}, f_2)$ is the target's state predictor after knowledge transfer. Considering the constraints given in (2.37)

$$f_1(x_1(t) \mid \mathbf{d_1(t)}, f_2) \propto f_1(y_1(t) \mid x_1(t)) f_1(x_1(t) \mid \mathbf{d_1(t\text{-}1)}) \chi(x_1(t) \in \mathbb{X}(t)). \tag{2.39}$$

$f_1(x_1(t) \mid \mathbf{d_1(t\text{-}1)})$ is the state predictor of an isolated filter. $\chi(x_1(t) \in \mathbb{X}(t))$ is the characteristic function (1.14) on set $\mathbb{X}(t)$ (2.37).

The characteristic function $\chi(x_1(t) \in \mathbb{X}(t))$ represents the constraints given be the intersection in (2.37) in other words the characteristic function $\chi(x_1(t) \in \mathbb{X}(t))$ represents the transferred knowledge from the source filter. After knowledge transfer the follows standard data update (2.2).

The knowledge transfer takes place between time update and data update. For better grasp of the complex estimation algorithm a brief summary is offered.

2.3 Algorithmic summary

Algorithm 1 FPD-KT

```
1: System Initialization:
     • set the initial time $t = 1$ and choose final time $\bar{t}$
     • set parameter values $A, B, C_1, C_2$
     • set prior values $\underline{x}_1(1), \overline{x}_1(1)$ for the target filter $f(x_1(1) \mid d_1(0)) = f(x_1(1)) = \mathcal{U}(\underline{x}_1(1), \overline{x}_1(1))$
     • set prior values $\underline{x}_2(1), \overline{x}_2(1)$ for the source filter $f(x_2(1) \mid d_2(0)) = f(x_2(1)) = \mathcal{U}(\underline{x}_2(1), \overline{x}_2(1))$
     • set the noise bounds $\nu, \omega_1, \omega_2$
2: Recursion:
   For $t \leftarrow 2$ to $\bar{t} - 1$
     (I) Knowledge transfer:
         Find $f_1(x_1(t) \mid \mathbf{d_1(t\text{-}1)}, f_2)$ according to (2.39) using (2.33) and constraints given in (2.37).
     (II) Data update:
         Process the local data on both filters. Compute $f(x_1(t) \mid \mathbf{d_1(t)})$ and $f(x_2(t) \mid \mathbf{d_2(t)})$ according to (2.8), (2.9). Perform the parallelotpic estimation as described in [28]. Circumscribe the parallelotope by an orthotope according to (2.14)-(2.17).
     (III) Time update:
         Perform data update on both filters. Compute $f(x_1(t) \mid \mathbf{d_1(t\text{-}1)})$ and $f(x_2(t) \mid \mathbf{d_2(t\text{-}1)})$ according to (2.21) and (2.9).
   End
3: Termination: set $t = \bar{t}$
     (I) Knowledge transfer:
         Find $f_1(x_1(\bar{t}) \mid \mathbf{d_1(\bar{t} - 1)}, f_2)$ according to (2.39) using (2.33) and constraints given in (2.37).
     (II) Data update:
         Process the local data on both filters. Compute $f(x_1(\bar{t}) \mid \mathbf{d_1(\bar{t})})$ and $f(x_2(\bar{t}) \mid \mathbf{d_2(\bar{t})})$ according to (2.8), (2.9). Perform the parallelotpic estimation as described in [28]. Circumscribe the parallelotope by an orthotope according to (2.14)-(2.17).
```

Chapter 3

Distributed information fusion

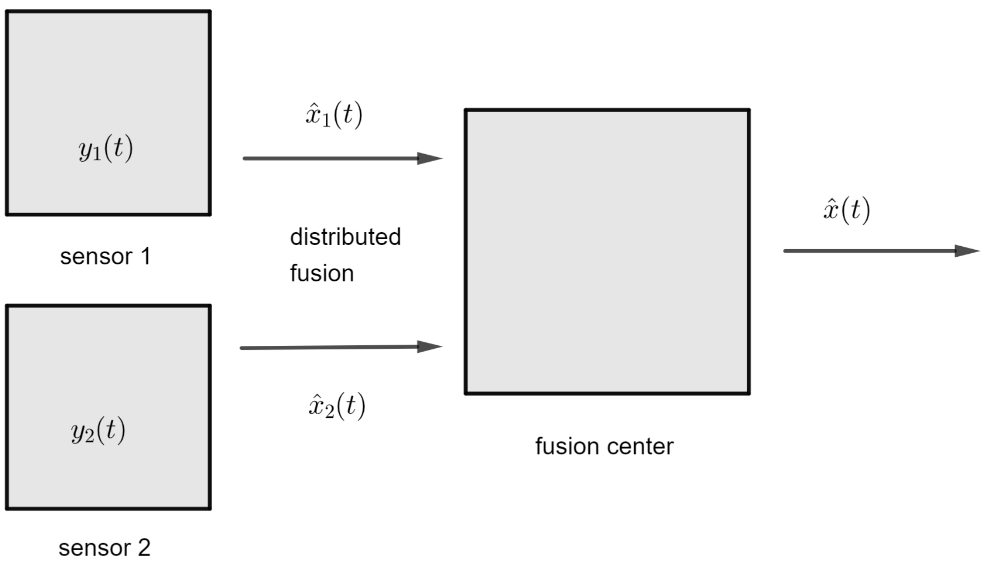


Figure 3.1: Distributed information fusion schema

Unless explicitly stated differently, the following chapter is based on work [7]. Distributed fusion, in contrast to knowledge transfer, does not distinguish between source and target filter. The sensors are on the same level, distinguishable by its uncertainties. The sensor measurements are processed locally and the result is a partial estimate derived from single sensor. Then, the local estimates are sent to the fusion center where the partial estimates are weighted according to the sensor information (precision etc.). The process produces a final state estimate at each time step t which includes information from all original sensors. See Figure 3.1.

For state estimation consider the state space model in (1.1) where $v(t)$ represents state noise and $w_j(t)$ output noise of the filter j, $j = 1, 2$. $v(t)$ and $w_j(t)$ are assumed to be of unknown distribution bounded by ellipsoids

$$\begin{aligned} \mathbb{V}(t) &= (v(t) : v(t)^T Q^{-1} v(t) \leq 1), \\ \mathbb{W}_j(t) &= (w_j(t) : w_j(t)^T R_j^{-1} w_j(t) \leq 1), \end{aligned} \tag{3.1}$$

where Q and R_j are positive definite known matrices .

The state estimate $\hat{x}$ is computed for each time step t using modified Kalman filter. Since Kalman filter was originally designed for systems with normally distributed noise the modification is necessary for considered system with known bounds but unknown distribution. Let the state estimation given by information from sensor j be called local state estimation and have form

$$\hat{x}_j(t) = A\hat{x}_j(t-1) + Bu(t-1) + K_j(y_j(t) - C_j A\hat{x}_j(t-1)), \tag{3.2}$$

where A and C_j are state space model parameter matrices from (1.1), $\hat{x}_j(t-1)$ is partial state estimate form previous time step $y_j(t)$ is the observation vector of sensor j and K_j is the Kalman gain matrix for sensor j.

Subsequently, let the final estimate $\hat{x}(t)$ be given by

$$\hat{x}(t) = \sum_{j=1}^{2} \Omega_j(t)\hat{x}_j(t), \tag{3.3}$$

where $\Omega_j(t)$ is the weighting matrix of the j-th sensor. The final estimate $\hat{x}(t)$ is the result of estimation fusion of all individual estimates $\hat{x}_j(t)$ from each sensor as depicted in the Figure 3.1.

The computation of Kalman gain K_j and weighting matrix of each sensor is presented in the following sections.

3.1 Kalman gain

According to [7], the optimal Kalman gain K_j in (3.2) is the result of following convex optimization problem

$$\min_{\vartheta_k > 0, P_j > 0, \theta_j > 0, K_j} \operatorname{tr}(\theta_j) \tag{3.4}$$

subject to

$$\begin{aligned} &\begin{bmatrix} -I & G_j A & M_j \\ A^T G_j^T & P_j & 0 \\ M_j^T & 0 & \theta_j \end{bmatrix} < 0 \\ &P_j - \vartheta_j I < 0 \\ &\vartheta_j < 1 \end{aligned} \tag{3.5}$$

where G_j and M_j are defined as

$$\begin{aligned} G_j &= I - K_j C_j, \\ M_j &= [G_j \quad -K_j]. \end{aligned} \tag{3.6}$$

The proof is found in [7].

Note that the Kalman gain matrix in [7] is considered time dependent. In this work, the Kalman gain does not evolve in time because the minimizing problem in considers only time non-dependent variables. The matrices A and C_j from (1.15) are time independent whereas in paper [7] the matrices A and C_j are considered time dependent.

After solving the above optimization problem, the partial state estimates $\hat{x}_j(t)$ of each sensor can be computed according to (3.12)

3.2 Weighting matrices

Since the Kalman gain and the state estimates $\hat{x}_j(t)$ of each sensor have been computed, the individual state estimate $\hat{x}_j(t)$ can be fused in the fusion center. For the individual estimate fusion are used weighting matrices Ω_j. The computation of the weighting matrices Ω_j is based on Mahalanobis distance according to [8].

Mahalanobis distance (MD) is usually used to identify outliers which occur due to uncertainties. Here, the MD is penalizing the sensor with higher uncertainty.

First, let define the output prediction $\gamma_j(t)$ as

$$\gamma_j(t) = C_j\hat{x}_j(t) \tag{3.7}$$

and the innovation covariance matrix

$$\Sigma_j = C_jPjC_j^T + D_j \tag{3.8}$$

which will be used for Mahalanobis distance (MD) computation.

In [8], the state and information noise in state-space model (1.1) was considered to be normally distributed with known covariance matrix D_j. In this work the noise has known bounds but unknown distribution therefore the computation formula was modified.

Firstly, the ellipsoid noise bounds must be approximated by an orthotope. The approximation is done as circumscription of en ellipsoid by an orthotope. The center of the ellipsoid (3.1) is a zero vector. For simplicity consider the ellipsoud matrix R_j diagonal. Then, the orthotope bounds in the i-th direction are $[-\sqrt{R_{j;ii}}, \sqrt{R_{j;ii}}]$ where $R_{j;ii}$ are diagonal elements of R_j.

Further, Gaussian approximation where mean an covariance are replaced by the first and second moment of the original distribution [20]. Then, the innovation covariance matrix is in form

$$\Sigma_j = C_jPjC_j^T + \frac{1}{3}R_j. \tag{3.9}$$

where R_j is the diagonal matrix form (3.1).

Mahalanobis distance measures the difference between the predicted value and its expected distribution while taking its variability into account. The predicted value is the output prediction $\gamma_j(t)$ and the expected value is the output ($y_j(t)$ the variability is expressed as the innovation covariance matrix Σ.

The Mahalanobis distance is defined as follows

$$M(y_j(t)) = \sqrt{(y_j(t) - \gamma_j(t))^T\Sigma^{-1}(y_j(t) - \gamma_j(t))}. \tag{3.10}$$

For computation purposes following approximation of MD from [25] will be used

$$M(y_j(t)) = \sum_{i=1}^{l_{y_j}} \left(\frac{(y_j(t) - \gamma_j(t))^2}{\Sigma_{j;ii}} \right), \tag{3.11}$$

where l_{y_j} is the length of output vector $y_j(t) = (y_{j;1}(t), \dots, y_{j;l_{y_j}}(t))$. $\Sigma_{j;ii}$ symbolizes diagonal elements of covariance matrix Σ_j.

The MD values are not used as weight values directly in order to soften the transitions because of the fluctuations of individual filters. Following sigmoid function is chosen to compute the weight

$$w_j = \frac{1}{1 + \exp\{-M(y_j(t))\}}, \tag{3.12}$$

The weighting matrices are then in form

$$\Omega_j(t) = w_j(t)I, \tag{3.13}$$

where $w_j(t)$ is a scalar computed in (3.12) and I is an identity matrix of size $l_x \times l_x$ and l_x is the length of the state vector x.

Finally, the global estimate $\hat{x}(t)$ is computed via (3.3).

3.3 Algorithmic summary

Algorithm 2 DF

```
1: System Initialization:
     • set the initial time $t = 1$ and choose final time $\bar{t}$
     • set parameter values $A$, $B$, $C_1$, $C_2$
     • set positive definite matrices $Q$, $R_1$ and $R_2$ defining the ellipsoids (3.1)
     • set the initial values $\hat{x}_1(1)$ and $\hat{x}_2(1)$
     • perform optimization (3.4) with respect to (3.5) to obtain $K_1$, $K_2$ and $P_1$, $P_2$
2: Recursion:
   For $t \leftarrow 2$ to $\bar{t}$
   Compute local estimates $\hat{x}_1(t)$ and $\hat{x}_2(t)$ according to (3.2).
   Obtain output prediction $\gamma_1(t)$ and $\gamma_2(t)$ using (3.7).
   Find weighting matrices $\Omega_1(t)$ and $\Omega_2(t)$ via (3.9), (3.11), (3.12) and (3.13)
   Weight estimates from both sensors via (3.3) to find global state estimate $\hat{x}(t)$.
   End
```

Chapter 4

Centralised information fusion

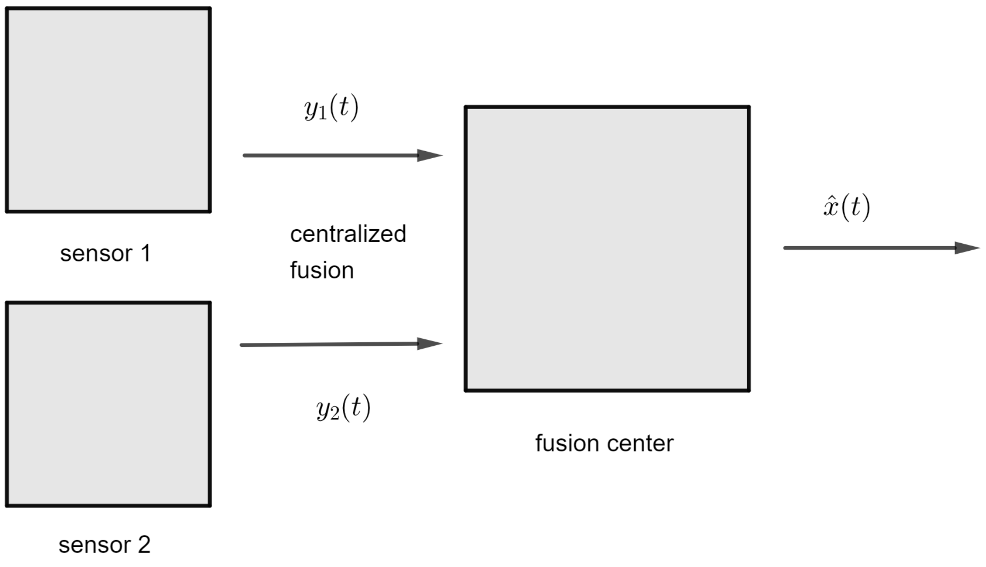


Figure 4.1: Centralised information fusion schema

The following chapter is based on [27]. The centralized information fusion is an estimation process which puts together low-dimensional vectors of individual measurements to form a single high-dimensional vector Figure 4.1. This single high-dimensional vector is then processed. The state estimation is divided into two main steps. The prediction step which can be compared to the data update (2.2) and the fusion update step which can be compared to the time update (2.3) from Chapter 2.

Consider the state space model in (1.1) where $v(t)$ represents state noise and $w_j(t)$ output noise of the filter j, $j = 1, 2$. $v(t)$ and $w_j(t)$ are assumed to be of unknown distribution bounded by ellipsoids (3.1) where Q and R_j are positive definite known matrices.

Let assume that the initial state $x(1)$ lies in ellipsoid

$$\mathbb{X}(1) = \{x(1) : (x(1) - \hat{x}(1))^T P^{-1}(1)(x(1) - \hat{x}(1)) \leq \sigma(1)\}, \tag{4.1}$$

where $\hat{x}(1)$ is the center of ellipsoid $\mathbb{X}(1)$ and $P(1)$ is positive definite matrix. The variable $\sigma(1)$ is a positive scalar variable, $\sigma(1) > 0$. Q, R_j and $P(1)$ are matrices of appropriate dimensions.

As in the two previous case, assume two sensors and two sets of output data $y_1(t)$ and $y_2(t)$ 4.1. The noise is uniformly distributed on a bounded support. The supports are in shape of an ellipsoids (3.1). The goal is to find the smallest possible set which contains the unknown state $x(t)$ $\forall t \in \mathbb{T}$.

The ellipsoid containing the state $x(t-1)$ is in form

$$\mathbb{X}(t-1) = \{x(t-1) : (x(t-1) - \hat{x}(t-1))^T P^{-1}(t-1)(x(t-1) - \hat{x}(t-1)) \leq \sigma(t-1)\}, \tag{4.2}$$

Combining equation (1.1), (3.1) and (4.2) results in

$$x(t) \in A\mathbb{X}(t-1) \oplus \mathbb{V}(t-1), \tag{4.3}$$

$$x(t) \in \{x(t-1) + v(t-1) : x(t-1) \in \ A\mathbb{X}(t-1), v(t-1) \in \mathbb{V}(t-1)\}. \tag{4.4}$$

The set (4.4) which results from the Minkowski sum contains the state $x(t)$. The ellipsoid which contains the state $x(t)$ must contain the whole set (4.4)

$$\mathbb{X}(t \mid t-1) \supseteq \{x(t-1) + v(t-1) : x(t-1) \in \ A\mathbb{X}(t-1), v(t-1) \in \mathbb{V}(t-1)\}, \tag{4.5}$$

The ellipsoid

$$\mathbb{X}(t \mid t-1) = \{x(t) : (x(t) - \hat{x}(t \mid t-1))^T P^{-1}(t \mid t-1)(x(t) - \hat{x}(t \mid t-1)) \leq \sigma(t \mid t-1)\}, \tag{4.6}$$

is the result of the prediction step.

Since the goal of prediction step is set, the aim of the fusion update step can be stated.

From equations (1.15) and (3.1) can be derived that $x(t)$ lies in the set

$$\mathbb{F}(t) = \{x(t) : (y_j(t) - C_j x(t))^T R_j^{-1}(y_j(t) - C_j x(t)) \leq 1, j = 1, 2\}, \tag{4.7}$$

The fusion update step puts together the information about $x(t)$ from prediction step in form of ellipsoid $\mathbb{X}(t \mid t-1)$ and the information given by the measurements on both filters in form of set $\mathbb{F}(t)$. The ellipsoid resulting from fusion update state is in form

$$\mathbb{X}(t) \supseteq \mathbb{X}(t \mid t-1) \cap \mathbb{F}(t). \tag{4.8}$$

Prediction step: The recursion starts with prediction step given the information about the state at previous time step $t-1$, $x(t-1) \in \mathbb{X}(t-1)$ where $\mathbb{X}(t-1)$ is from (4.2) and $w(t-1) \in \mathbb{K}(t-1)$ where $\mathbb{K}(t-1)$ is from (3.1). Following the first equation of the state-space model (1.15)

$$\begin{aligned}
\hat{x}(t|t-1) &= A\hat{x}(t-1) + Bu(t-1) \\
\sigma(t|t-1) &= \sigma(t-1) \\
P(t|t-1) &= \left(1 + \frac{1}{p(t)}\right) AP(t-1)A^T + \frac{1 + p(t)}{\sigma(t|t-1)} Q
\end{aligned} \tag{4.9}$$

then the prediction step ends with $\forall p(t) \in (0,\infty)$, $x(t) \in \mathbb{X}(t \mid t-1)$.

The equations in (4.9) emerged from the Lemma 1 in [27]. The Lemma 1 states that two ellipsoids of the same dimension $\mathbb{Z}_{E_1} = \{z : (z-a_1)^T E_1 (z-a_1) \leq 1\}$ and $\mathbb{Z}_{E_2} = \{z : (z-a_2)^T E_2 (z-a_2) \leq 1\}$ where $z, a_1, a_2 \in \mathbb{R}^n$ and $E_1, E_1 \in \mathbb{R}^{n,n}$, then $\forall p(t) \in (0,\infty)$,

$$\mathbb{Z}_{E_3} = \{z : (z-a_3)^T E_1 (z-a_3) \leq 1\} \supseteq \mathbb{Z}_{E_1} \oplus \mathbb{Z}_{E_2} \tag{4.10}$$

where

$$\begin{aligned} a_3 &= a_1 + a_2 \\ E_3 &= \left(1 + \frac{1}{p(t)}\right) E_1 + (1+p(t)) E_2. \end{aligned} \tag{4.11}$$

In (4.9) for $\mathbb{Z}_{E_1}$ is taken $\mathbb{X}(t-1)$ (4.2) with center $\hat{x}(t-1)$ and ellipsoid matrix $P(t-1)$, for $\mathbb{Z}_{E_2}$ is taken $\mathbb{V}(t-1)$ (3.1) with center zero vector and ellipsoid matrix Q, then $\mathbb{Z}_{E_3}$ corresponds with $\mathbb{X}(t \mid t-1)$ (4.6) with center $\hat{x}(t-1 \mid t)$ and ellipsoid matrix $P(t-1 \mid t)$.

In contrast to [27], this thesis considers state space model (1.1) with control input vector $u(t)$ and input matrix B. The result $Bu(t)$ is a known vector. In order to integrate the input vector to the estimation algorithm, the center $\hat{x}(t-1 \mid x)$ on an ellipsoid $\mathbb{X}(t \mid t-1)$ (4.6) is calculated according to (4.9). The center $\hat{x}(t-1 \mid t)$ calculation in (4.9) differs from center calculation in [27]. The center $\hat{x}(t-1 \mid x)$ in (4.9) is shifted by the controlled input $Bu(t)$.

Fusion update step:

$$\begin{aligned} \hat{x}(t) &= \hat{x}(t|t-1) + q(t)P(t)C^T R^{-1} \mu(t) \\ \sigma(t) &= \sigma(t|t-1) + q(t) - q(t)(\mu(t))^T \left(q(t)CP(t|t-1)C^T + R\right)^{-1} \mu(t) \\ P^{-1}(t) &= P^{-1}(t|t-1) + q(t)C^T R^{-1} C \end{aligned} \tag{4.12}$$

where

$$\begin{aligned} \mu(t) &= y(t) - C\hat{x}(t|t-1), \\ y(t) &= [y_1(t)^T, y_2(t)^T]^T, \\ v(t) &= [v_1(t)^T, v_2(t)^T]^T, \\ C &= [C_1^T, C_2^T]^T, \\ R^{-1} &= \operatorname{diag}\left(\alpha_j(t) R_j^{-1}\right), \end{aligned} \tag{4.13}$$

and $\alpha_j(t) \in [0,1]$, $\sum_{j=1}^{2} \alpha_j(t) = 1$, $\forall q(t) \in [0,\infty)$, $x(t) \in \mathbb{X}(t) \supseteq \mathbb{X}(t \mid t-1) \cap \mathbb{F}(t)$. For proof see [27]. The state estimate $\hat{x}(t)$ is the center of the ellipsoid $\mathbb{X}(t)$.

The second equation in (4.13) implies that the dimension of the fused output vector $y(t)$ results from the sum of dimension of the two output vectors $y_1(t)$ from sensor 1 and $y_2(t)$ from sensor 2, see Figure 4.1.

4.1 Optimal parameters

In prediction and fusion update step occurred multiple parameters. The goal of the estimation is to find the smallest possible ellipsoid which contains the state $x(t)$. The smaller the set containing the state the more accurate is the state estimate. The parameters can be optimized to minimize the bounding ellipsoid, to secure stable estimation error or can be set to a constant.

In prediction step Lemma 1 from [27] is applied and the the result is (4.2) which is true $\forall p(t) \in (0, \infty)$. To avoid complex nonlinear equations, the parameter $p(t)$ from prediction step is derived as follows

$$p_O(t) = \left(\frac{\sigma(t|t-1)\mathrm{tr}(AP(t-1)A^T)}{\mathrm{tr}(Q)} \right)^{\frac{1}{2}} \tag{4.14}$$

where $p_O(t)$ is the optimal value of $p(t)$. The parameter $q(t)$ from fusion update step is optimized to ensure stability of the error estimate. The optimal value $q_O(t)$ of $q(t)$ is the result of minimizing task

$$q_O(t) = \arg \min_{q(t) \geq 0} f(q(t)) \tag{4.15}$$

where $f(q(t)) = \sigma(t)$. When $\mu(t)^T R^{-1} \mu(t) \geq 1$ then for the $q_O(t)$ hods true equation

$$\mu(t)^T (R + q_O(t) C P(t \mid t-1) C^T)^{-1} R (R + q_O(t) C P(t \mid t-1) C^T)^{-1} \mu(t) = 1. \tag{4.16}$$

For $\mu(t)^T R^{-1} \mu(t) < 1$, there is no solution for the equation above (4.16). The optimal value of parameter $q(t)$, $q_O(t) = 0$.

Note that if $q(t) = 0$ then in fusion update step (4.12) the equations are

$$\begin{aligned} \hat{x}(t) &= \hat{x}(t|t-1) \\ \sigma(t) &= \sigma(t|t-1) \\ P^{-1}(t) &= P^{-1}(t|t-1) \end{aligned} \tag{4.17}$$

This implies that if $q(t) = 0$, then no fusion update takes place. In the bounding sets sense, if $q(t) = 0$, then $\mathbb{X}(t) = \mathbb{X}(t \mid t-1)$.

The parameters $\alpha_j(t)$ are chosen at every time step in order to increase tolerance to faulty outputs $y_j(t)$.

$$\alpha_j(t) = \frac{\left(\left\| \mu_j(t) \right\|_{R_j^{-1}} \right)^{-1}}{\sum_{j=1}^{2} \left(\left\| \mu_j(t) \right\|_{R_j^{-1}} \right)^{-1}} \tag{4.18}$$

where $||\cdot||_{R_j}$ is a matrix norm

$$\left\| \mu_j(t) \right\|_{R_j^{-1}} = \left(\mu_j(t)^T R_j^{-1} \mu_j(t) \right)^{\frac{1}{2}} \tag{4.19}$$

and $\mu_j(t) = y_j(t) - C_j \hat{x}(t|t-1)$.

The following section summarizes the state estimation via centralized fusion.

4.2 Algorithmic summary

Algorithm 3 CF

```
1: System Initialization:
     • set the initial time $t = 1$ and choose final time $\bar{t}$
     • set parameter values $A$, $B$, $C_1$, $C_2$
     • set prior values $\hat{x}(1)$, $P(1)$ and $\sigma(1)$ for the initial state (4.1)
     • set positive definite matrices $Q$, $R_1$ and $R_2$ defining the ellipsoids (3.1).
2: Recursion:
   For $t \leftarrow 2$ to $\bar{t}$
     (I) Prediction:
         Compute parameter $p(t)$ according to (4.14). Obtain $\hat{x}(t|t-1)$, $\sigma(t|t-1)$ and $P(t|t-1)$ using (4.9).
     (II) Fusion update:
         Compute parameters $q(t)$, $\alpha_1(t)$, $\alpha_2(t)$ according to (4.18), (4.16). Fuse data from both sensors via (4.13). Find $\hat{x}(t)$, $\sigma(t)$ and $P(t)$ using (4.12).
   End
```

Chapter 5

Experiments

In this section, experiment setups are described and experiment results are presented. For experimental comparison of the three methods for information transfer described in previous sections multiple parameter setting is employed.

Firstly, kinematic model is used. The kinematic model is derived from equations of motion with constant velocity. The model is more generic than equations of motion since it considers the third order derivatives of position not zero but zero mean processes [6]. The deterministic kinematic model would consider the third order derivatives of position $z(t)$ in generic coordinates to be zero

$$\dddot{z}(t) = 0. \tag{5.1}$$

The continuous Wiener process acceleration model considers the third order derivatives of position $z(t)$ in generic coordinates to be white noise $v(t)$

$$\dddot{z}(t) = v(t) \tag{5.2}$$

with zero mean

$$\mathrm{E}[v(t)] = 0 \tag{5.3}$$

The acceleration is considered a Wiener process. The state $x(t)$ from (4.12) is then in form

$$x(t) = \begin{bmatrix} z(t) \\ \dot{z}(t) \\ \ddot{z}(t) \end{bmatrix}. \tag{5.4}$$

The discrete state equation (first equation in (1.1)) of the position-velocity-acceleration model [6] is

$$x(t) = \begin{bmatrix} 1 & T_0 & 0.5T_0 \\ 0 & 1 & T_0 \\ 0 & 0 & 1 \end{bmatrix} x(t-1) + \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} u(t-1) + v(t), \tag{5.5}$$

where T_0 is the sampling period and input matrix B is considered a zero matrix.

The measurement equations (second equation in (1.1)) depend on the sensor output. If the sensors measure only position, then the measurements equation are in form

$$y_j(t) = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} x(t) + w_j(t). \tag{5.6}$$

If the sensors measure position and velocity, then the measurements equation are in form

$$y_j(t) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} x(t) + w_j(t). \tag{5.7}$$

If the sensors measure position, velocity and acceleration, then the measurements equation are in form

$$y_j(t) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} x(t) + w_j(t) = , \tag{5.8}$$

where $j = 1, 2$.

For following experiments consider sensors measuring position and velocity (5.7). The model matrices are

$$A = \begin{bmatrix} 1 & T_0 & 0.5T_0 \\ 0 & 1 & T_0 \\ 0 & 0 & 1 \end{bmatrix} \otimes I_2, \quad B = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \otimes I_2, \quad C_1 = C_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \otimes I_2, \tag{5.9}$$

where T_0 is the length of sampling period and $T_0 = 0.1$. The one dimensional system was extended into two dimensional system. The extension was achieved by Kronecker product with a two by two dimensional unit matrix I_2. The Kronecker product was applied on the state parameter matrix as well as on the output parameter matrices. The resulting state parameter matrix A is of double size in both dimensions i.e. $A \in \mathbb{R}^{6\times6}$. The output parameter matrices also double its size in both dimensions therefore C_1 and $C_2 \in \mathbb{R}^{4\times6}$.

The state $x(t)$ that is to be estimated is a vector of six components

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \\ x_5(t) \\ x_6(t) \end{bmatrix}, \tag{5.10}$$

where $x_1(t)$ and $x_2(t)$ are the position coordinates of the tracking object at time t, $x_3(t)$ and $x_4(t)$ are two dimensional velocity coordinates at time t and $x_5(t)$, $x_6(t)$ are acceleration coordinates of the tracked object at time t.

For further experiment a linear system of second-order was chosen. The system is commonly used to describe many dynamic process [10]. In a physical second-order system, energy is stored in two different elements which exchange the stored energy. In mechanical systems, it is the exchange between mass and stiffness, in electrical systems, the exchange takes place between capacitors and inductors and in hydraulic systems the energy exchanges between fluid inertance and capacitance. Following linear differential equation describes the system

$$a_2\ddot{z} + a_1\dot{z} + a_0 z = b_0 u \tag{5.11}$$

The response generated by the system given by (5.11) is either non-oscillatory decaying or continuous oscillatory depending on the parameters a_0, a_1 and a_2 [10].

Let consider the non-oscillatory version with parameters $a_0 = 1$, $a_1 = 2$ and $a_2 = 1$. After transformation into the state space model and discretization with sampling period $T_0 = 0.1$ the system has following form

$$A = \begin{bmatrix} 0.8144 & -0.0905 \\ 0.0905 & 0.9953 \end{bmatrix}, \quad B = \begin{bmatrix} 0.0905 \\ 0.0047 \end{bmatrix}, \quad C_1 = C_2 = \begin{bmatrix} 0 & 1 \end{bmatrix}. \tag{5.12}$$

Then, the state $x(t)$ that is to be estimated is a vector of two components

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}. \tag{5.13}$$

For performance comparison of the three above mentioned algorithms for state estimation in two-sensor system the total norm-squared error was used. The total norm-squared error (TNSE) is defined as follows

$$\mathrm{TNSE}_i = \sum_{t=1}^{\bar{t}} (\hat{x}_i(t) - x_i(t))^2. \tag{5.14}$$

where x stands for either simulated state values or real state data and $\hat{x}$ is state estimation.

Let be defined the estimation error $e(t)$ as

$$e_i(t) = \hat{x}_i(t) - x_i(t), \tag{5.15}$$

Algorithms from Chapters 2, 3 and 4 were implemented in MATLAB to perform all further described experiments. The MATLAB scripts with implemented algorithms and recorded data are loaded on a CD which is attached to this thesis.

Further, the algorithms described in Chapters 2, 3 and 4 are referred to as FPD-KT, DF and CF. The algorithm for state estimation via FPD without knowledge transfer from [16] is the part of FPD-KT described in Section 2.1 and denoted FPD-IF.

To solve minimizing task (3.4), a MATLAB LMI Toolbox function `mincx` was used.

The input data $u(t)$ were randomly generated from Gaussian distribution with zero mean and variance=1 using build-in MATLAB function `randn`.

For all following experiments, the parameter $q(t)$ from (4.12) was set $q(t) = 1$. The parameters ϑ_1, ϑ_2 from (3.5) were set $\vartheta_1 = \vartheta_2 = 0.5$. The prior values $P(1)$ and $\sigma(1)$ for the initial state (4.1) were set to $P(1) = I$ and $\sigma(1) = 1$.

The first state estimation algorithm FPD-KT has noise support bounded by orthotopes (1.10) and the second two consider noise to be bounded by an ellipsoid (1.13). To unify the noise bounds approach for computation purposes, let assume the orthotopic noise bounds v, ω_1 and ω_2 from (2.4) which form the orthotopes edges to be also the ellipsoids bounds. All noise bounds v, ω_1 and ω_2 are assumed to be equal in all directions i.e., all the orothotpe's edges are of the same length hence the corresponding ellipsoid are spheres with diameter values equal to noise bounds v^2, ω_1^2 and ω_2^2 The ellipsoids are given by the matrices Q, R_1 and R_2 (3.1). The relation between orthotopic and elliptic bounds setting is following

$$\begin{aligned} Q &= v^2 I, \\ R_1 &= \omega_1^2 I, \\ R_2 &= \omega_2^2 I, \end{aligned} \tag{5.16}$$

where I is an identity matrix. Then, the orthotopes defined by v, ω_1 and ω_2 circumscribe the ellipsoids given by Q, R_1 and R_2.

In some graphs, a different marker is used to distinguish between two lines that overlap.

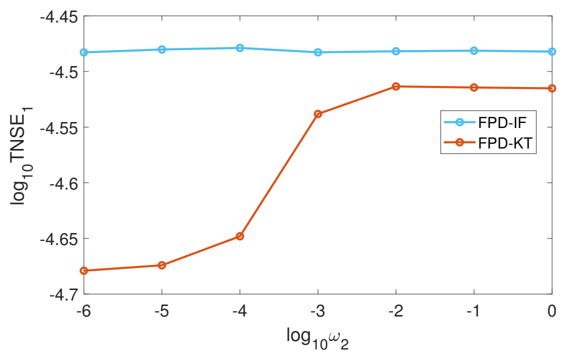
5.1 Simulated data

Simulation was implemented in MATLAB. To generate uniform noise a build-in MATLAB function `unifrnd` was used. All further presented results for simulated data were averaged over 500 independent runs.

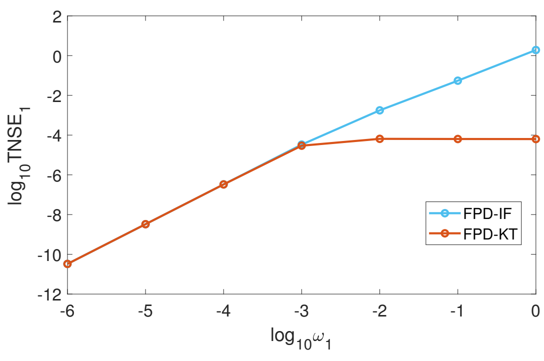
For all experiments with simulated data following parameter setting was applied. Prior values for FPD-KT bounds were set to $\overline{x}_1(1) = 0.5\mathbf{j}_{l_x}$, $\underline{x}_1(1) = -0.5\mathbf{j}_{l_x}$, $\overline{x}_2(1) = 0.05\mathbf{j}_{l_x}$ and $\underline{x}_2(1) = -0.05\mathbf{j}_{l_x}$ where $\mathbf{j}_{l_x}$ is a vector of ones of length l_x. The initial values for DF and CF were set to $\hat{x}_1(1) = \hat{x}_2(1) = \hat{x}(1) = \mathbf{0}_{l_x}$ where $\mathbf{0}_{l_x}$ is a zero vector of length l_x.

Isolated filter and knowledge transfer

In order to validate the contribution of knowledge transfer, the state estimation results of an isolated (single) filter algorithm FPD-IF from [16] and two-sensor algorithm for knowledge transfer FPD-KT described in Chapter 2 are compared.



(a) TNSE (5.14) of the first component of the state vector (5.10) with $\omega_1 = 10^{-3}$ and changing second (source) filter precision



(b) TNSE (5.14) of the first component of the state vector (5.10) with $\omega_2 = 10^{-3}$ and changing first (target) filter precision.

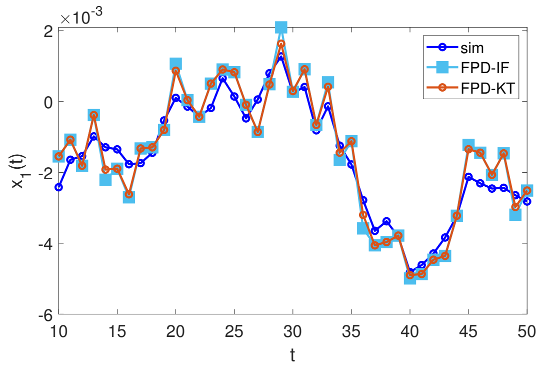
Figure 5.1: Comparison of FPD-KT and FPD-IF for $\nu = 10^{-3}$, $\bar{t} = 100$ and parameter matrices system (5.9).

Figure 5.1 compares the performance of the state estimation of an isolated filter (FPD-IF) and state estimation with knowledge transfer between two filters (FPD-KT) using parameter matrices setting (5.9). Both axes are in logarithmic scale for higher transparency. Note that in Figure 5.1a, the changing precision of the second filter does not influence the TNSE of the FPD-IF estimate. On the other hand, the TNSE of the FPD-IF estimate increases with decreasing precision of the second filter.

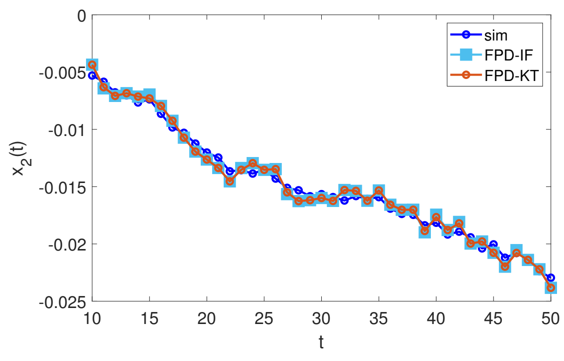
In Figure 5.1b, the TNSE of the FPD-KT estimate rises until the precision of the first filter reaches the precision of the source filter. Then, the TNSE keeps almost constant value similar to the TNSE values of the FPD-KT on the right side of the graph in the Figure 5.1a. The TNSE of the FPD-IF estimates increases with decreasing target filter precision.

Figure sequence 5.2 depicts the development of simulated states and state estimates for time interval $t \in [10, 50]$. Successively, the values for all state vector components (5.10) are displayed in graphs in Figure 5.2. The estimates done by FPD-KT and FPD-IF overlap in the Figure 5.2. Both estimates are simillar to he simulated states in Figures 5.2a-d. In Figures 5.2e-f, the estimates have constant zero value while the simulation is non-zero.

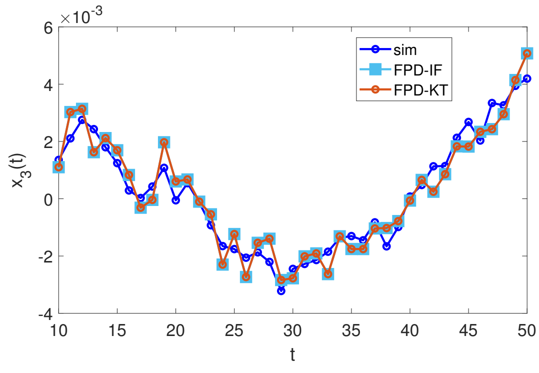
The estimation via FPD-IF shows slightly higher TNSE values than the FPD-KT estimation in Figure 5.3. For lower uncertainty of the second sensor, the FPD-KT performs better in TNSE-sense (Figure 5.3a). In Figure 5.3b, the difference in TNSE for higher uncertainties is significantly bigger. FPD-KT shows almost the same TNSE values in both Figures 5.3a and 5.3b. Figure 5.4 compares the state estimation and state simulation for time interval $t \in [10, 50]$ for the two components of the state vector (5.13). Both estimates and simulation overlap in Figure 5.4a as well as in Figure 5.4b.



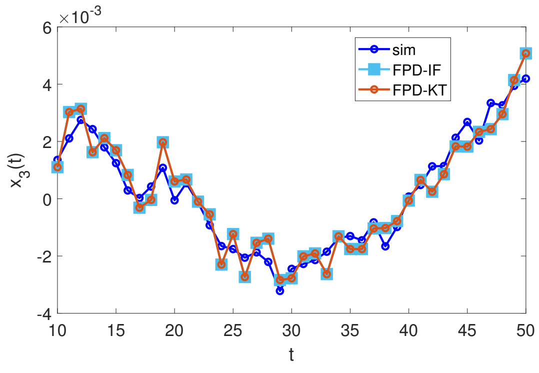
(a)



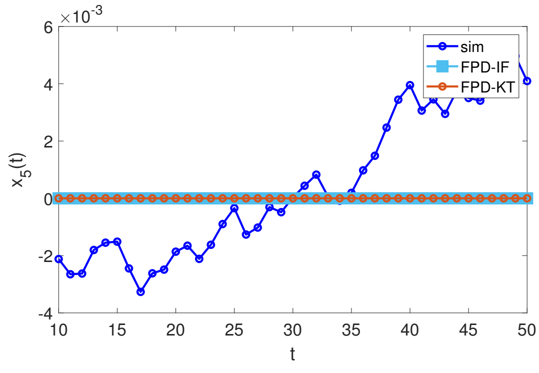
(b)



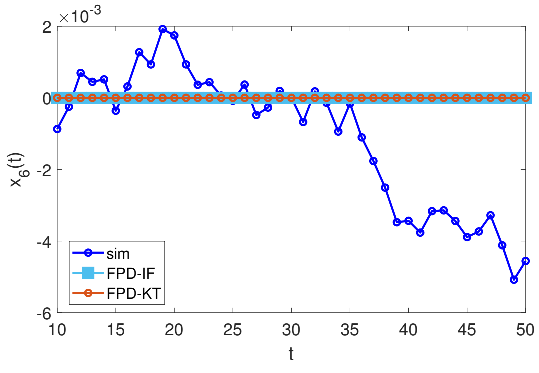
(c)



(d)

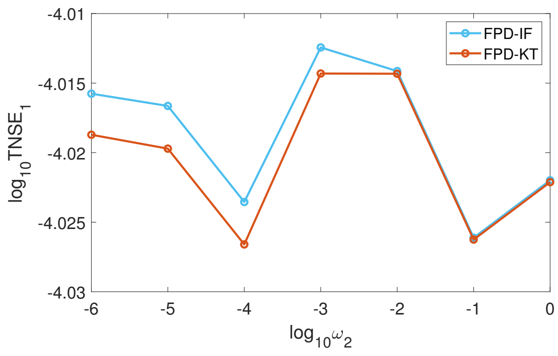


(e)

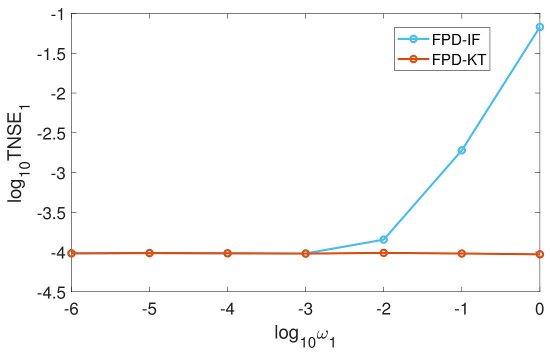


(f)

Figure 5.2: State estimates (FPD-KT and FPD-IF) and simulation (sim) with the noise setting $\omega_1 = \omega_2 = \nu = 10^{-3}$ for each component of the state vector (5.10) and parameter matrices system (5.9).

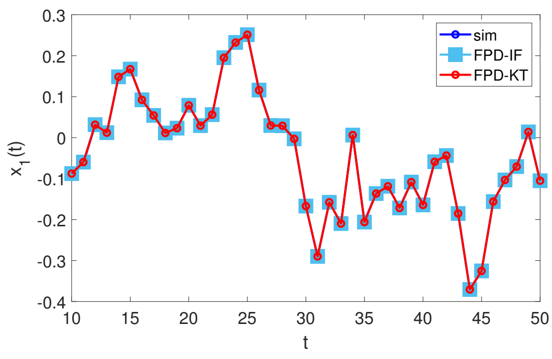


(a) TNSE (5.14) of the first component of the state vector (5.13) with $\omega_1 = 10^{-3}$ and changing second (source) filter precision.

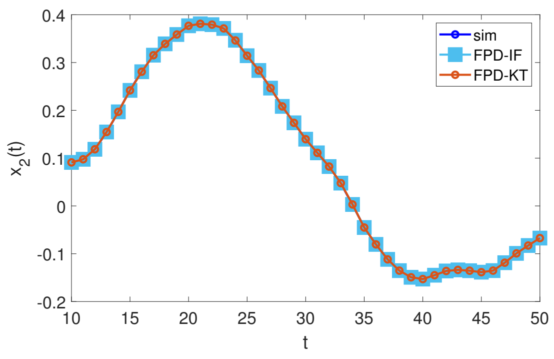


(b) TNSE (5.14) of the first component of the state vector (5.13) with $\omega_2 = 10^{-3}$ and changing first (target) filter precision.

Figure 5.3: Comparison of FPD-KT and FPD-IF for $\nu = 10^{-3}$, $\bar{t} = 100$ and parameter matrices system (5.12).



(a)

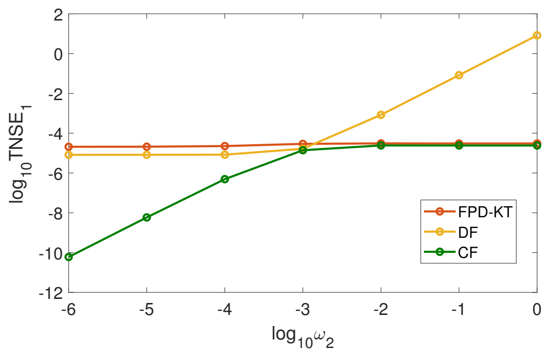


(b)

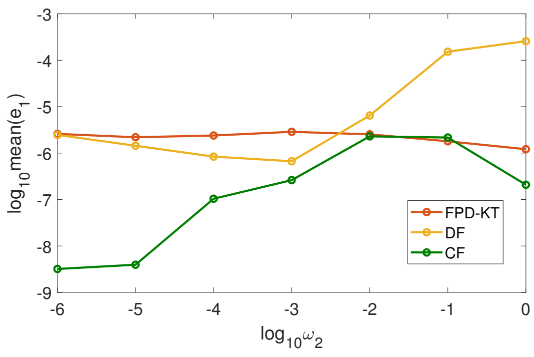
Figure 5.4: State estimates (FPD-KT and FPD-IF) and simulation (sim) with the setting $\omega_1 = \omega_2 = \nu = 10^{-3}$ for each component of the state vector (5.13) and parameter matrices system (5.12).

Knowledge transfer and information fusion

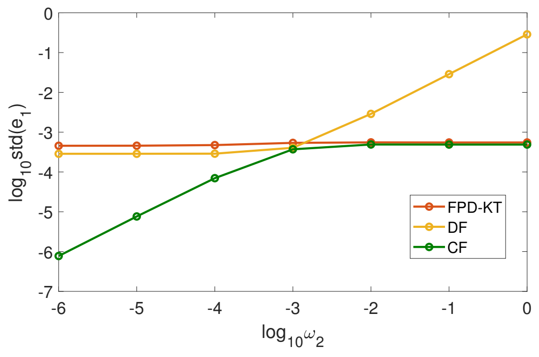
Now, the method for state estimation using knowledge transfer FPD-KT is compared to the two methods using information fusion DF and CF.



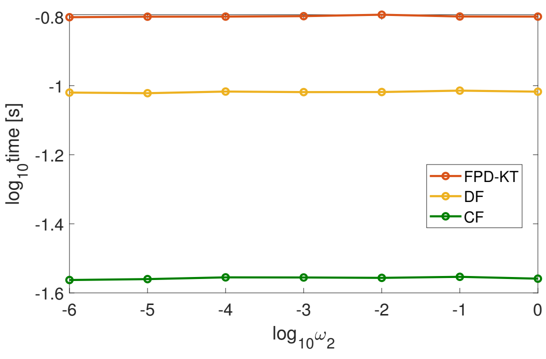
(a) TNSE (5.14) of the first component of the state vector (5.10).



(b) Absolute value of the mean of the estimation error (5.15) of the first component of the state vector (5.10).



(c) Standard deviation of the estimation error (5.15) of the first component of the state vector (5.10)



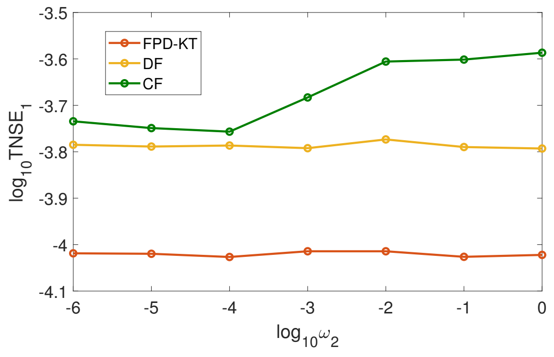
(d) Duration of the computation in time [s].

Figure 5.5: Comparison of FPD-KT, DF and CF for $\omega_1 = \nu = 10^{-3}$ and $\bar{t} = 100$ with changing second (source) filter precision and parameter matrices system (5.9).

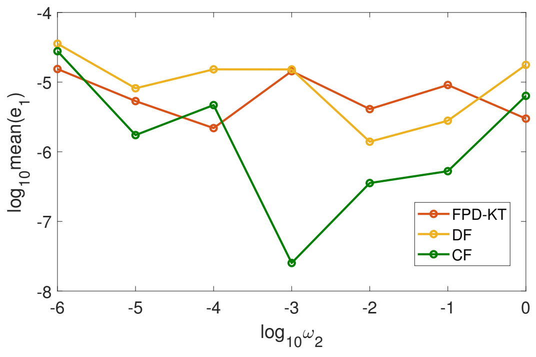
The results of state estimation in Figure 5.5 compare the performance three methods for state estimation which combine information from two sensors. ω_2 is the uncertainty of the source filter for FPD-KT Figure 2.2. For the two information fusion methods DF and CF, ω_2 is the uncertainty of the second sensor Figure 3.1, Figure 4.1, respectively. Results in Figure 5.5 are based on experiment computed for parameter matrices system (5.9) and time sequence of length $\bar{t} = 100$.

All four criteria in Figure 5.5 show almost constant performance of the FPD-KT. CF performs significantly better than DF and FPD-KT for lower uncertainties of the second sensor. TNSE, mean and std of DF is eminently higher than TNSE, mean and std of CF and FPD-KT for higher uncertainties of the second sensor. See Figures 5.5a, 5.5b and 5.5c. Computation time of all methods shows no relation to the second sensor precision, see Figure 5.5d.

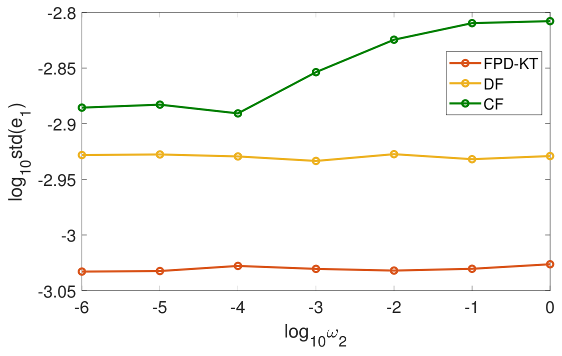
Figure 5.6 demonstrates results of state estimation on parameter matrices system (5.12). Lowest TNSE and std values are assigned to the FPD-KT method while the performance is almost constant with changing second (source) filter precision, see Figures 5.6a and 5.6c. DF also shows constant performance in TNSE- and std-sense although higher then FPD-KT. Further, CF shows poorer performance for higher uncertainties of the second sensor. The mean values in Figure 5.6b of all methods are comparable, CF



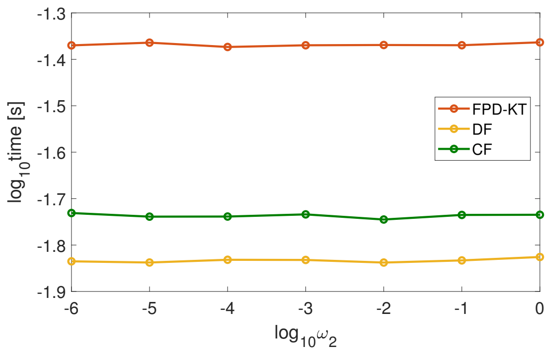
(a) TNSE (5.14) of the first component of the state vector (5.10).



(b) Absolute value of the mean of the estimation error (5.15) of the first component of the state vector (5.10).



(c) Standard deviation of the estimation error (5.15) of the first component of the state vector (5.10).



(d) Duration of the computation in time [s].

Figure 5.6: Comparison of FPD-KT, DF and CF for $\omega_1 = \nu = 10^{-3}$ and $\bar{t} = 100$ with changing second (source) filter precision and parameter matrices system (5.12).

shows the highest fluctuations. FPD-KT has highest computation time demands, DF is slightly less time-demanding than CF. Time demands of all methods are constant, see 5.6d.

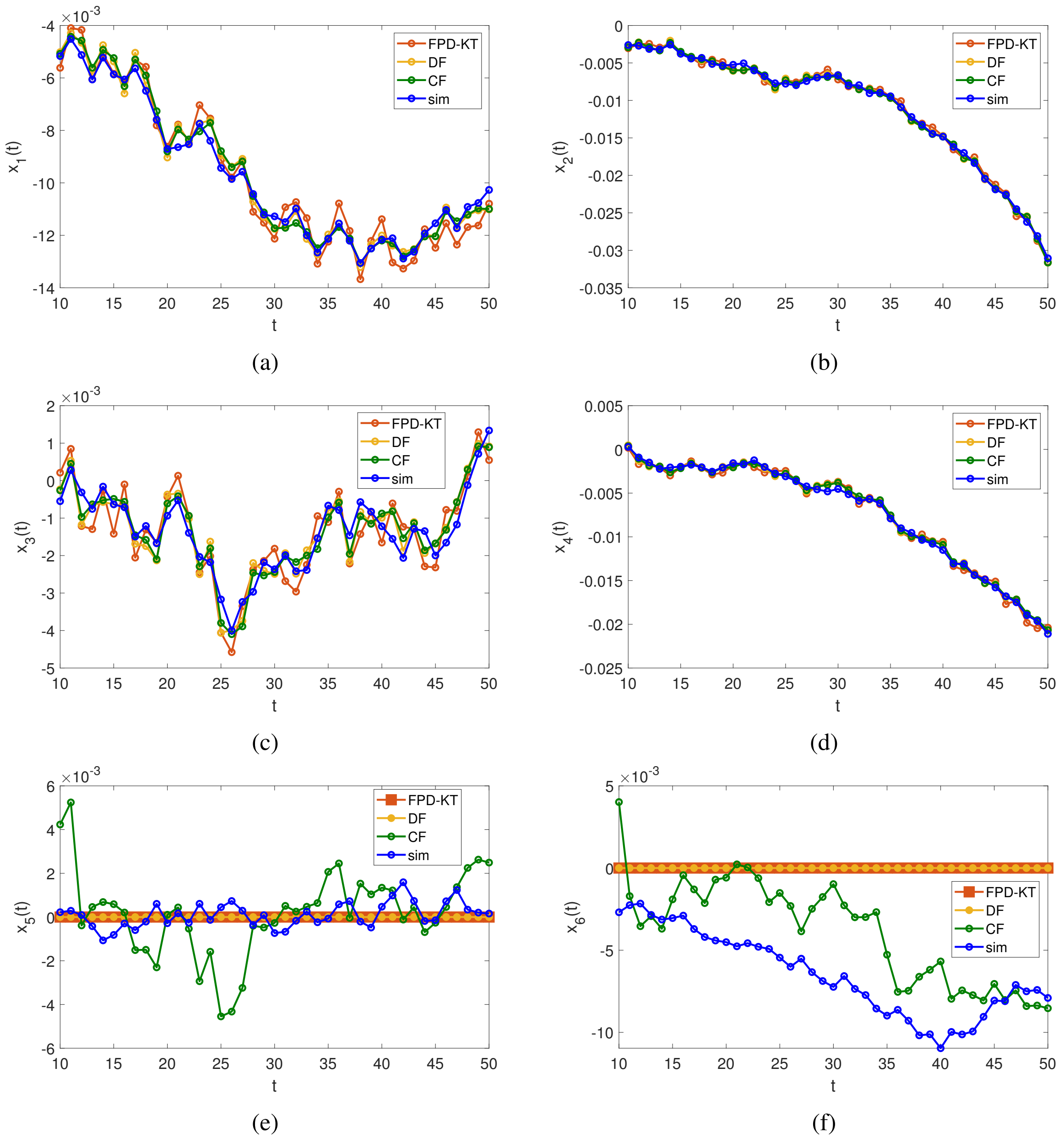
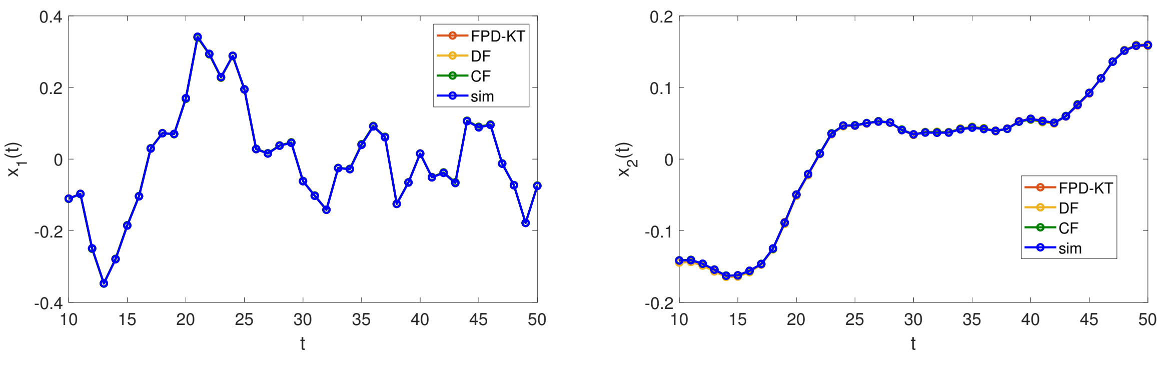


Figure 5.7: State estimates (FPD-KT, DF and CF) and simulation (sim) with the setting $\omega_1 = \omega_2 = \nu = 10^{-3}$ for each component of the state vector (5.10) and parameter matrices system (5.9).

Figures 5.7 are graphical demonstration of simulated states and estimated states values of each component of the state vector (5.10) and parameter matrices system (5.9). The estimation of the first four components of the state vector (5.10) turns out likewise for all methods, see Figures 5.7a-d. The estimation of the two remaining components (Figures 5.7e-f) which stand for acceleration is less successful. The FPD-KT and DF estimate is a constant zero while CF gives the only non-zero estimate.

The demonstration of simulated states and estimated states values of each component of the state vector (5.13) and parameter matrices system (5.12) is shown in Figures 5.8. On the given time interval



(a) (b)

Figure 5.8: State estimates (FPD-KT, DF and CF) and simulation (sim) with the setting $\omega_1 = \omega_2 = \nu = 10^{-3}$ for each component of the state vector (5.13) with parameter matrices system (5.12).

$t \in [10, 50]$, the simulation overlaps with all estimates.

Although all experiments with simulated data were averaged over 500 Monte Carlo runs, there are still some visible fluctuations, Figures 5.1, 5.3, 5.5 and 5.6.

5.2 Real data

Experiments with real data were performed for parameter system (5.9). To collect real data on position, speed and location, an app called AndroSensor available on Google Play [4] was used. The app was chosen because it records all required information and saves records into .csv file. Most apps record only position and speed hence the acceleration data are missing and also many apps do not save the data in convenient formate. AndroSensor records data about location, acceleration, speed, orientation, accuracy, magnetic field, time and others.

In order to collect data from two sensors, AndroSensor was installed to two different smartphones. Data were recorded on both smartphones simultaneously during one car trip. In the app settings, the Speed Unit was changed from default kilometres per hour to meters per second and the Accelerometer coords were set to world coords instead of device coords.

Before the state estimation experiments with real world data could be performed, the raw data from AndroSensor were preprocessed. The app saves data record into .csv file. The .csv file generated by the app was unreadable in MATLAB. The data file was first opened in RStudio and then saved as .csv file. After this modification MATLAB was able to read the file as table. Next, some data needed to be transformed. The data about location were recorded in spatial coordinate system [1] hence the data were transformed into Cartesian coordinate system using following transformation

$$\begin{aligned} x_1(t) =& R \sin(\text{lat}(t)) \cos(\text{lon}(t)), \\ x_2(t) =& R \sin(\text{lat}(t)) \sin(\text{lon}(t)), \end{aligned} \tag{5.17}$$

depicted in Figure 5.9. R stands for volumetric mean radius of Earth, according to [2], $R = 6,371,000$ m, lat(t) and lon(t) stand for latitude and longitude at time t, $x_1(t)$ and $x_2(t)$ are position coordinates on the X (horizontal) and Y (vertical) axes at time t i.e., first and second component of the state vector (5.10).

The volumetric mean radius of Earth R is in the order of millions of meters hence the computed position data are also in the order of millions of meters. For better comprehensibility of the values, it

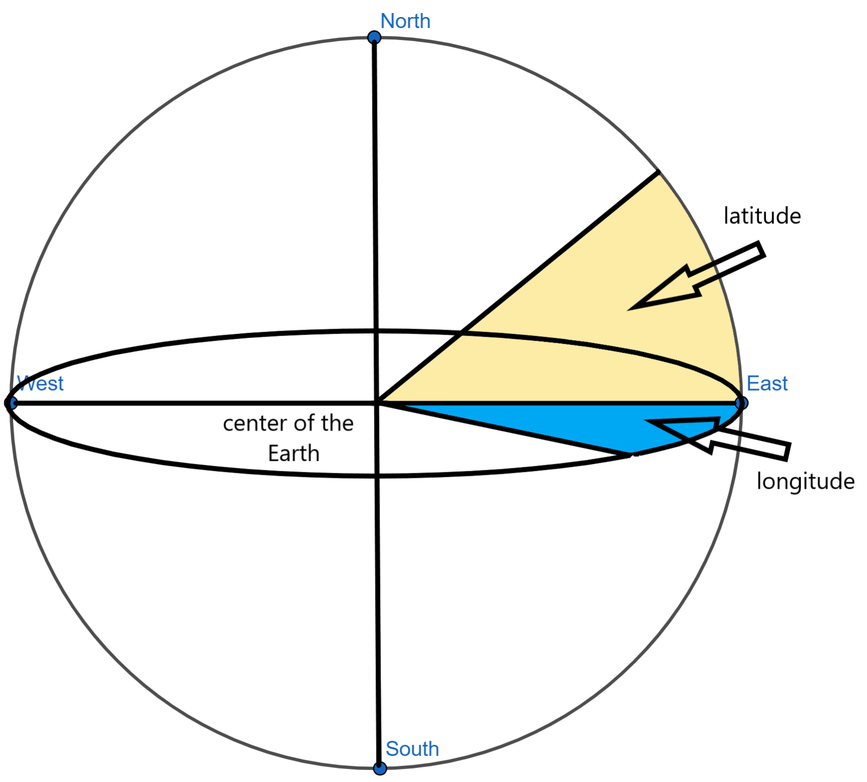


Figure 5.9: Latitude and longitude

was decided to move the starting point to zero i.e., $x_1(1) = x_2(1) = 0$. Then, $x_1(t)$ and $x_2(t)$, $\forall t > 1$, state the distance from the point at time t to zero (starting point) in meters.

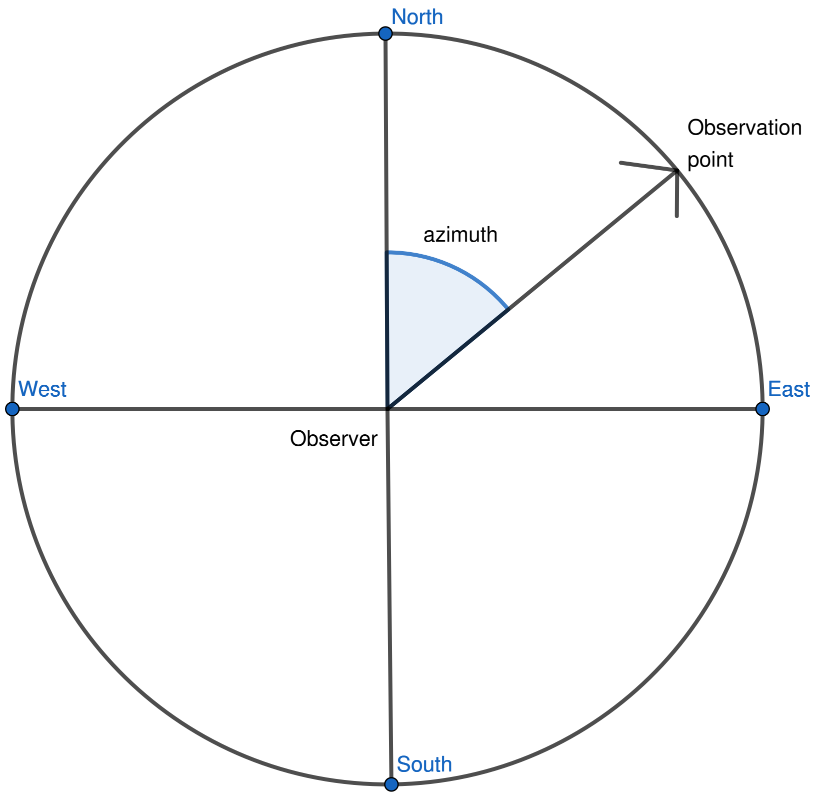


Figure 5.10: Azimuth

Speed was recorded by the app only as speed vector. For the estimation task, the speed values in direction of the X axis and Y axis are needed. To decompose the speed vector into two parts another information is required. This information is azimuth which is defined as the angle created by the observation point, the observer and the reference point (North). The azimuth is measured from North clockwise se Figure 5.10 [5]. In this case, the azimuth is the angle between the speed vector and geographical North. the decomposition of the speed vector into X axes and Y axes using azimuth is depicted in the picture 5.11, then the transformation is in form

$$\begin{aligned} x_3(t) =& \text{speed}(t)\sin(\text{azimuth}(t)),\\ x_4(t) =& \text{speed}(t)\cos(\text{azimuth}(t)), \end{aligned} \tag{5.18}$$

where speed(t) stands for the recorded speed vector at time t, azimuth(t) stands for azimuth at time t, $x_3(t)$ (speed X) and $x_4(t)$ (speed Y) stands for speed in the direction of X and Y axes at time t i.e., third and fourth component of the state vector (5.10).

All data stating angles (latitude, longitude, azimuth) are recorded in degrees. In order to solve equations (5.17) and (5.18) in MATLAB, angle records in radians need to be converted into radians. The conversion from degrees to radians goes as follows

$$\alpha_r = \frac{\pi\alpha_d}{180}, \tag{5.19}$$

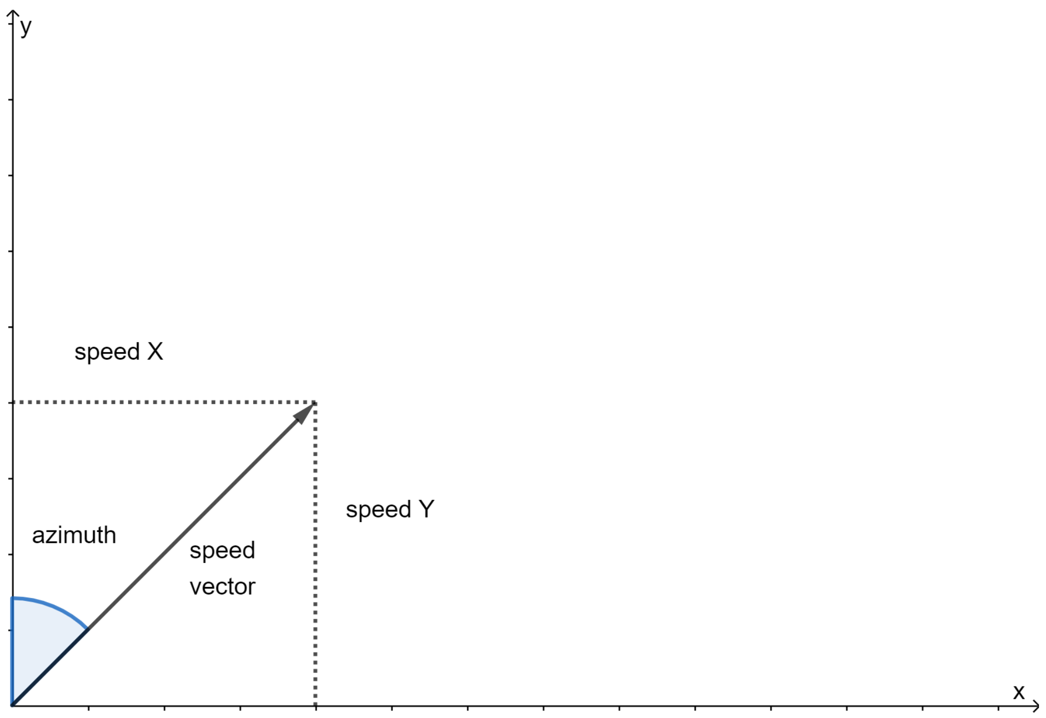


Figure 5.11: Azimuth and speed vector

where α_r stands for angle in radians and α_d for angle in degrees.

Since the real recorded data position-speed-acceleration are now two-dimensional data, for state estimation the parameter matrices (5.9) was used. The estimated states were be compared to the measured (real) data.

The estimation was performed according to the algorithmic summaries in Sections 2.3, 3.3 and 4.2 of the three described state estimation methods. For input data $y_1(t)$ and $y_2(t)$ were taken the measured data about position and speed. For noise bounds v, ω_1 and ω_2 were chosen recorded accuracy data. Accuracy of both sensor was measured during the whole time sequence. For the estimation purposes, for noise bounds v, ω_1 and ω_2 were taken means of the recorded accuracy data. The matrices Q, R_1 and R_2 defining the ellipsoids (3.1) were computed according to (5.17).

The recorded accuracies were 3.4 and 9.4 in meters. The sensor with accuracy=3.4 is from now on called the more accurate sensor. The sensor with accuracy=9.4 is from now on called the less accurate sensor.

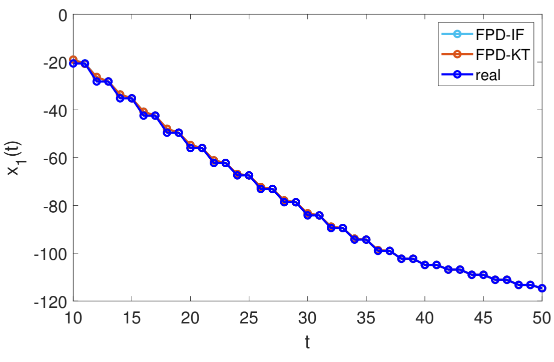
For all experiments with real data, following initial parameter setting was used. The prior values for FPD-KT bounds $\overline{x}_1(1)$, $\overline{x}_2(1)$, $\underline{x}_1(1)$ and $\underline{x}_2(1)$ were set to the position, speed and acceleration values at time $t = 1$ of the chosen target sensor $+0.5\mathbf{j}_{l_x}$, $-0.5\mathbf{j}_{l_x}$, respectively. The initial values of DF $\hat{x}_1(1)$, $\hat{x}_2(1)$ and CF $\hat{x}(1)$ were set to the position, speed acceleration values at time $t = 1$ of the chosen target sensor.

The real data (real) are the measured data about position, speed and acceleration by the sensor which is chosen as the target. Further, there are experiments with the less accurate sensor as target as well as with the more accurate sensor as target. The estimated data are then compared to the measured data by the sensor selected as the target.

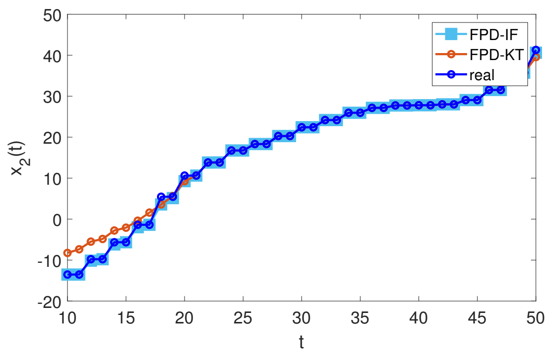
The chosen length of the time is 2000 time steps.

Isolated filter and knowledge transfer

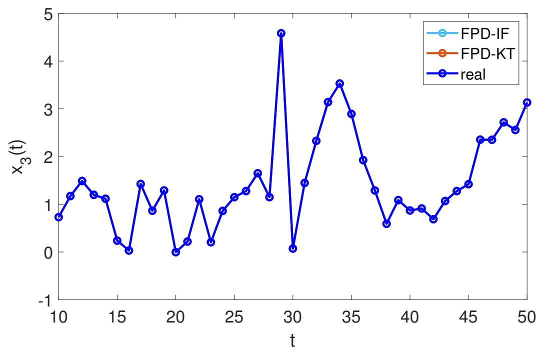
Performance of an isolated (single) filter state estimation algorithm FPD-IF from [16] using real data is compared to the two-sensor algorithm for knowledge transfer FPD-KT described in Chapter 2.



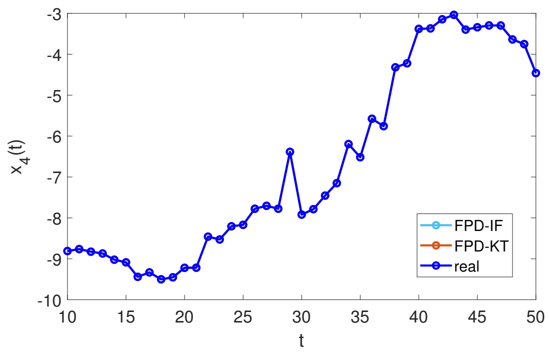
(a)



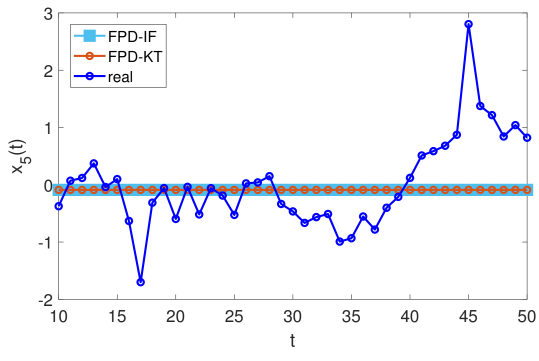
(b)



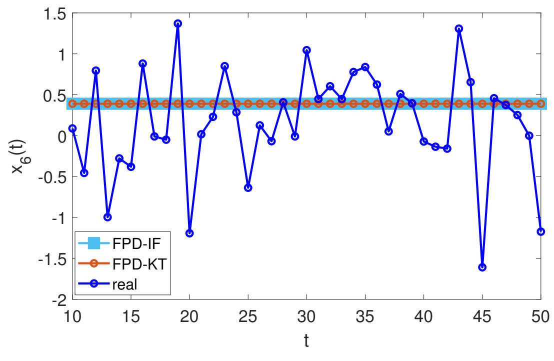
(c)



(d)



(e)



(f)

Figure 5.12: State estimates (FPD-KT and FPD-IF) and real data (real) with the more accurate sensor as target for each component of the state vector (5.10).

On a chosen interval $t \in [10, 50]$, the recorded real data and state estimates by FPD-KT and FPD-IF almost bled together with the exception of acceleration data, Figure 5.12. The acceleration is estimated as constant by the method with and also without knowledge transfer.

Knowledge transfer and information fusion

State estimation FPD-KT with knowledge transfer was already compared to a single-source filtration FPD-IF. Further, the knowledge transfer in FPD-KT is compared to the centralised information fusion (CF) and to the distributed information fusion (DF).

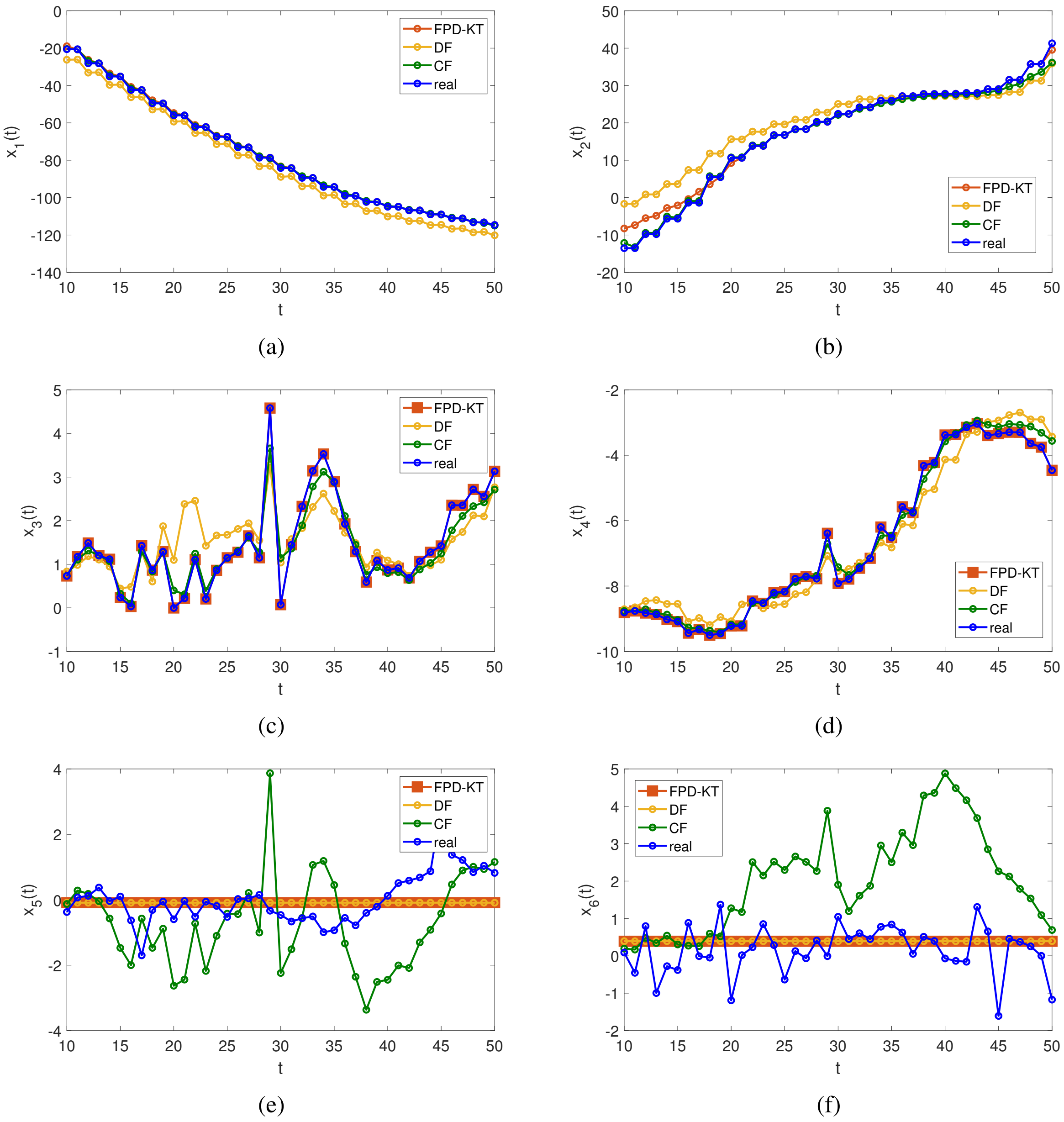


Figure 5.13: State estimates (FPD-KT, DF and CF) and real data (real) with the more accurate sensor as target for each component of the state vector (5.10).

Based on Figures 5.13a-d, the FPD-KT and CF show the best performance in estimation of the real data. The only non-constant acceleration estimates gives the CF method, see Figures 5.13e-f.

Since the distinction between FPD-IF and FPD-KT is hard to compare in Figures 5.12 and 5.13, the Tables 5.1 and 5.2 are presented. All values in the Tables 5.1 and 5.2 were rounded to two significant

Table 5.1: TNSE (5.14), mean, median, standard deviation (std) of the state estimation error (5.15) and computation time for $\bar{t}$ = 2000 with the more accurate sensor as target.

	FPD-IF	FPD-KT	DF	CF
TNSE$_1$	3.8×10^{3}	4.5×10^{3}	2.9×10^{7}	5.3×10^{5}
mean($e_1(t)$)	-5.4×10^{-1}	-4.6×10^{-1}	7.0×10^{1}	-1.2×10^{1}
median($e_1(t)$)	0	0	9.3	-1.1×10^{1}
std($e_1(t)$)	1.3	1.4	9.8×10^{1}	1.1×10^{1}
TNSE$_2$	1.8×10^{4}	1.3×10^{5}	3.8×10^{7}	1.2×10^{6}
mean($e_2(t)$)	1.2	4.4	-8.0×10^{1}	1.5×10^{1}
median($e_2(t)$)	0	1.8×10^{-12}	-1.0×10^{1}	9.2
std($e_2(t)$)	2.8	6.9	1.1×10^{2}	2.0×10^{2}
TNSE$_3$	7.3×10^{-25}	5.7×10^{-26}	9.2×10^{5}	3.2×10^{5}
mean($e_3(t)$)	1.2×10^{-15}	4.0×10^{-15}	1.8×10^{1}	9.9
median($e_3(t)$)	0	0	2.4×10^{1}	1.0×10^{1}
std($e_3(t)$)	1.9×10^{-14}	1.6×10^{-14}	1.2×10^{1}	7.9
TNSE$_4$	6.7×10^{-25}	7.0×10^{-25}	3.0×10^{5}	1.5×10^{5}
mean($e_4(t)$)	4.2×10^{-16}	-1.4×10^{-15}	-6.9	-5.0
median($e_4(t)$)	0	0	-1.5	-1.9
std($e_4(t)$)	1.8×10^{-14}	1.8×10^{-14}	1.0×10^{1}	7.2
TNSE$_5$	1.2×10^{3}	1.2×10^{3}	1.2×10^{3}	1.9×10^{4}
mean($e_5(t)$)	-2.9×10^{-1}	-2.9×10^{-1}	-2.9×10^{-1}	4.4×10^{-1}
median($e_5(t)$)	-2.9×10^{-1}	-2.9×10^{-1}	-2.9×10^{-1}	4.8×10^{-1}
std($e_5(t)$)	7.3×10^{-1}	7.3×10^{-1}	7.3×10^{-1}	3.1
TNSE$_6$	2.3×10^{3}	2.3×10^{3}	2.3×10^{3}	2.1×10^{3}
mean($e_6(t)$)	-5.8×10^{-1}	-5.8×10^{-1}	-5.7×10^{-1}	-5.6×10^{-1}
median($e_6(t)$)	-5.6×10^{-1}	-5.6×10^{-1}	-5.7×10^{-1}	-8.9×10^{-1}
std($e_6(t)$)	9.0×10^{-1}	9.0×10^{-1}	9.0×10^{-1}	3.2
computation time [s]	1.5×10^{-2}	2.4×10^{-2}	8.1×10^{-2}	2.4×10^{-1}

figures.

Based on TNSE, mean, median and std of the estimation error and computation time, all four methods are compared, for results see Tables 5.1 and 5.2.

Firstly, the more accurate sensor was selected as target Table 5.1. The estimated states are compared to the states measured by target in Table 5.1. Note, that FPD-IF and FPD-KT differ only a little. The other two estimation methods DF and DF show poorer results in all components. The results are the same for FPD-IF, FPD-KT and DF in the fifth and the sixth component. This is the consequence of the constant-value estimates for acceleration.

Secondly, the less accurate sensor was chosen as target. The data measured by the less accurate sensor are considered the real data. The estimates are compared to the real data in Table 5.2. Here, the results of FPD-IF and FPD-KT differ, especially in the first two components. The result for acceleration are the same for FPD-IF, FPD-KT and DF because all these methods estimate constant acceleration. The results for DF and CF show higher estimation error.

Figures 2.2, 2.3, 3.1, 4.1, 5.9, 5.10 and 5.11 were plotted in GeoGebra [3]. Figure 2.1 was plotted in Paint 3D. The remaining figures were plotted in MATLAB.

Table 5.2: TNSE (5.14), mean, median, standard deviation (std) of the state estimation error (5.15) and computation time for $\bar{t}$ = 2000 with the less accurate sensor as target.

	FPD-IF	FPD-KT	DF	CF
TNSE$_1$	4.8×10^{-21}	2.6×10^{3}	3.0×10^{7}	6.8×10^{4}
mean($e_1(t)$)	-4.5×10^{-13}	-3.2×10^{-1}	7.1×10^{1}	-1.1
median($e_1(t)$)	0	0	8.7	1.4×10^{-1}
std($e_1(t)$)	1.4×10^{-12}	1.1	1.0×10^{2}	5.7
TNSE$_2$	5.7×10^{-21}	1.3×10^{3}	4.2×10^{7}	3.9×10^{5}
mean($e_2(t)$)	5.9×10^{-13}	1.2×10^{-2}	-9.0×10^{2}	-6.0
median($e_2(t)$)	0	0	-2.0×10^{1}	-8.7
std($e_2(t)$)	1.6×10^{-12}	8.1×10^{-1}	1.1×10^{2}	1.3×10^{1}
TNSE$_3$	5.3×10^{-24}	4.2×10^{-24}	9.3×10^{5}	2.7×10^{7}
mean($e_3(t)$)	-2.2×10^{-15}	-8.8×10^{-15}	-1.8×10^{1}	-3.0×10^{1}
median($e_3(t)$)	0	0	-2.3×10^{1}	-4.0×10^{1}
std($e_3(t)$)	5.1×10^{-14}	4.5×10^{-14}	1.2×10^{1}	2.0×10^{1}
TNSE$_4$	5.0×10^{-24}	5.2×10^{-24}	3.0×10^{5}	8.5×10^{5}
mean($e_4(t)$)	-8.9×10^{-16}	4.2×10^{-15}	7.1	1.2×10^{1}
median($e_4(t)$)	0	0	1.8	2.4
std($e_4(t)$)	5.0×10^{-14}	5.1×10^{-14}	1.0×10^{1}	1.7×10^{1}
TNSE$_5$	3.2×10^{3}	3.2×10^{3}	3.2×10^{3}	4.6×10^{4}
mean($e_5(t)$)	-1.1	-1.1	-1.2	-1.1
median($e_5(t)$)	-1.1	-1.1	-1.1	-8.8×10^{-1}
std($e_5(t)$)	5.1×10^{-1}	5.1×10^{-1}	5.1×10^{-1}	4.7
TNSE$_6$	7.9×10^{2}	7.9×10^{2}	7.9×10^{2}	4.7×10^{4}
mean($e_6(t)$)	-2.9×10^{-1}	-2.9×10^{-1}	-2.9	-1.3
median($e_6(t)$)	-2.8×10^{-1}	-2.8×10^{-1}	$-2.\times10^{-1}$	-1.4
std($e_6(t)$)	5.6×10^{-1}	5.6×10^{-1}	5.6×10^{-1}	4.7
computation time [s]	6.1×10^{-1}	9.4×10^{-1}	7.9×10^{-2}	3.0×10^{1}

Chapter 6

Discussion

The FPD-KT, DF and CF estimation performance was examined, using simulated data with uniform noise and using real data. The collection and processing of real data was described earlier in this work.

The CF method shows the best results for estimating the state vector for parameter matrices setting (5.9). CF has significantly lower TNSE, mean and std for lower uncertainties of the second sensor (Figure 5.5). This proves the positive transfer. Also the TNSE and std stop rising when the precision of the second sensor reaches the precision of the first sensor. After this turning point, TNSE and std do not rise with rising uncertainty of the second sensor. That proofs the absence of negative transfer. On the other hand, DF algorithm shows (in Figure 5.5) exactly opposite results. DF prevents the positive transfer (constant values of TNSE and std for low uncertainties) and allows the negative transfer (rising TNSE and std for increasing uncertainties).

In simulation experiments, the time expense of all methods is higher for system (5.9) than for (5.12) (compare Figure 5.5d and 5.6d). In (5.9), the state vector has higher dimension than in (5.12). Hence all corresponding parameter matrices used during computation are of higher dimension. This all prolongs the computation time.

Note that all methods except CF estimate the last two components of the state vector (5.10) which stand for acceleration as constant zero, Figures 5.2 and 5.7. The estimate is not necessarily zero. It depends on the initial value of acceleration at the begging of estimation sequence at time $t = 1$. This is proven in real data experiments where the initial values are non-zero, see Figure 5.12f and 5.13f. Here, the estimates are constant but non-zero. This proved that FPD-IF, FPD-KT and DF is unable to update the initial values of acceleration. As already mentioned, the CF estimates the acceleration and updates the initial acceleration value. The reason is unclear. The main difference between the CF method and the other methods is, that all the other methods keep the same dimension of states and output during the whole estimation process. The CF method processes high-dimensional vector composed of all sensor measurements and then estimates the states. The ability of CF to update acceleration might lie in the high-dimensional estimation process. This hypothesis could be subject to further analysis.

The system (5.12) was applied to test the estimation methods. Firstly, the system (5.12) has non-zero input matrix B comparing to (5.9). The results for experiments on system (5.12) show that all methods work for non-zero inputs. Secondly, the results in Figures 5.4 and 5.8 show that all components of the state vector (5.13) are estimated correctly. Hence the problem with constant-value estimation is specific to the system (5.9).

The FPD-KT and FPD-IF are compared to assess the contribution of knowledge transfer to state estimation method from [16]. The contribution of FPD-KT lies in allowing positive transfer and preventing negative transfer. The difference in TNSE values for FPD-KT is small thus invisible in Figure 5.5. On the other hand, it is conveniently demonstrated in Figure 5.1. The TNSE of FPD-KT estimates is lower for smaller uncertainties of the source but when the target and source precision are equal the TNSE of FPD-KT stops rising. The TNSE in 5.1a is constant for FPD-IF because the isolated filtration has no information about the second source since it processes only the target data. Hence the isolated filtration can not be influenced by the source filter precision.

Figure 5.1b also proves the positive contribution of knowledge transfer in state estimation via FPD. Firstly, FPD-IF and FPD-KT keep the same performance for lower uncertainties of the first sensor. This means, that if the target filter has higher precision, the FPD-KT acts like an isolated filter. If the target filter has lower accuracy, then the TNSE of FPD-KT stops rising thanks to the information from the source which corrects the target information. The isolated filter TNSE in Figure 5.1b rises constantly since the only available information is the target information with decreasing accuracy.

The results in Table 5.1 confirm the results displayed in graphs in Figure 5.12. The estimates done by FPD-IF and FPD-KT are very similar. The result of FPD-IF match the real data even more than the FPD-KT results, see Figure 5.12b. The results in Table 5.1 indicate that the isolated filter method estimates better despite of the fact that the simulation results show the opposite. In simulation experiments, there was the simulated state to compare with the estimates. In real data experiments, there are no objectively accurate state data available. The only available data are those measured by the sensors which are also used as inputs.

If the estimated data are compared to the data obtained by the sensor considered as target, then the FPD-IF performs better, see Tables 5.1 and 5.2. The FPD-IF processed only data from the target and then compares the estimates to the target data. The FPD-KT process data from both sensors (target and source) and then compares the estimates to the target data. Hence the FPD-KT estimates differ from the target data more than the FPD-IF estimates.

The Table 5.2 demonstrates higher discrepancy in the FPD-IF and FPD-KT estimates. The experiment with results in Table 5.2 was conducted choosing the less accurate sensor as target. Here, the FPD-KT estimates for the first two components of the state vector are very different from the FPD-IF. The reason is, that FPD-KT as well as the other methods (DF and CF) with two-sensor based estimation penalize the data obtained from the less accurate sensor. The lesser accurate sensor the more penalized the received data hence the estimate differs more from data from less accurate sensor. If the estimates are compared to the data from the less accurate sensor, then they must differ.

Tables 5.1 and 5.2 show that the computation time of FPD-KT is longer than the computation time of FPD-IF. The core of both algorithms is the same Section 2.1. In FPD-KT, there are two parallel filtrations and a knowledge transfer step. In FPD-IF, there is only one filtration which is exactly the same as one of the filtrations in FPD-KT and there is no knowledge transfer step. The FPD-KT extends FPD-IF hence the FPD-KT method is more time demanding.

Based on real data experiments, it is not possible to decide whether or not the FPD-KT performs better than FPD-IF. On the other hand, the difference in results between FPD-KT and FPD-IF in Tables 5.1. and 5.2 can be viewed as proof that the knowledge transfer also works on real data.

To compare performances the simulation experiments are more suitable. The real data experiments could be extended by the use of more sensors with different precision.

Conclusion

In this Master's Thesis, the state estimation method based on knowledge transfer via fully probabilistic design FPD-KT for linear systems with bounded noise was described and analysed. FPD-KT was compared to the state estimation without knowledge transfer FPD-IF to asses the benefits of knowledge transfer from the second sensor.

Further, two methods for state estimation for linear systems with bounded noise which use modified Kalman filter and fuse information from two sensors to obtain better estimates were described and analysed. Distributed information fusion performs partial state estimation on both sensors, then the local estimates are fused in the fusion center. Centralised information fusion fuses raw information from sensors and performs state estimation in the fusion center. Finally, the FPD-KT was compared to DF and CF

To compare the above mentioned methods, experiments were prepared. Firstly, the experiments were conducted on simulated data with uniformly distributed noise. Secondly, real data on position, speed and acceleration were collected, processed and used form experiments.

The contribution of knowledge transfer was confirmed by the conducted experiments. Even though the results of experiments on real data show better performance of the isolated filter, the simulation experiments proved that FPD-KT managed to avoid negative transfer. It was discovered, that while comparing a single source and multiple source estimation method, the simulated data are more suitable for this purpose than the real data.

Centralised information fusion CF showed better estimation results on simulated data as well as on real data experiments comparing to other distributed information fusion DF. CF also managed to give non-zero estimate of the acceleration of the object while the other methods fail to estimate the acceleration. On the other hand, DF achieved to estimate at the lowest time expense. The FPD-KT showed the best performance in estimating real data in comparison to the information fusion methods DF and CF.

Experiments on real data could be extended. The output data would be recorded in the same way as described in this thesis, by the two independent sensors. The state vector would be recorded by an independent third sensor. Then, the estimates could be compared to the data from to third sensor thus avoiding the confusion in estimation performance and receiving more objective results.

Bibliography

[1] Coordinate system. `https://ltb.itc.utwente.nl/page/498/concept/81538`. [Online; accessed 19/4/2022].

[2] Earth fact sheet. `https://nssdc.gsfc.nasa.gov/planetary/factsheet/earthfact.html`. [Online; accessed 4/4/2022].

[3] Geogebra. `https://www.geogebra.org/calculator`. [Online; accessed 11/4/2022].

[4] Google play. `https://play.google.com/store/apps/details?id=com.fivasim.androsensor`. [Online; accessed 4/4/2022].

[5] Planar coordinate system. `https://ltb.itc.utwente.nl/page/498/concept/81590`. [Online; accessed 19/4/2022].

[6] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan. *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons, 2004.

[7] B. Chen, G. Hu, D. W. C. Ho, and L. Yu. A New Approach to Linear/Nonlinear Distributed Fusion Estimation Problem. *IEEE Transactions on Automatic Control*, 64(3):1301–1308, 2019.

[8] A. F. Echeverri, H. Medeiros, R. Walsh, Y. Reznichenko, and R. Povinelli. Real-time hierarchical Bayesian data fusion for vision-based target tracking with unmanned aerial platforms. In *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1262–1270, June 2018.

[9] C. Foley and A. Quinn. Fully probabilistic design for knowledge transfer in a pair of Kalman filters. In *IEEE Signal Processing Letters*, number 25, pages 487–490. 2018.

[10] B. Friedland. Control system design: An introduction to the state-space methods. *Courier corporation*, 2012.

[11] U. D. Hanebeck and J. Horn. Fusing Information Simultaneously Corrupted by Uncertainties with Known Bounds and Random Noise with Known Distribution. *Information Fusion*, 1(1):55 – 63, 2000.

[12] L. Jirsa, L. Kuklišová Pavelková, and A. Quinn. Bayesian transfer learning between uniformly modelled Bayesian filters. In *Informatics in Control, Automation and Robotics*, pages 151–168. Springer International Publishing, 2021.

[13] L. Jirsa, L. Pavelková, and A. Quinn. Knowledge transfer in a pair of uniformly modelled Bayesian filters. In *Proceedings of the 16th International Conference on Informatics in Control, Automation and Robotics - Volume 1: ICINCO,*, pages 499–506. INSTICC, SciTePress, 2019.

[14] M. Kárný. Towards fully probabilistic control design. In *Automatica*, volume 32, pages 1719–1722. 1996.

[15] M. Kárný and T. Kroupa. Axiomatisation of fully probabilistic design. In *Informatics in Control, Automation and Robotics (ICINCO 2018)*, volume 186, pages 388–394. 2012.

[16] E. Lainová. State estimation of constrained linear systems. *Bachelor's Thesis (Bc.). Czech Technical University in Prague. Faculty of Nuclear Sciences and Physical Engineering, Department of Mathematics*, 2019, in Czech.

[17] T. Meng, X. Jing, Z. Yan, and W. Pedrycz. A survey on machine learning for data fusion. *Information Fusion*, 57:115 – 129, 2020.

[18] S. J. Pan and Q. Yang. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, Oct 2010.

[19] M. Papež and A. Quinn. Robust Bayesian transfer learning between Kalman filters. In *IEEE Internation Workshop on Machine Learning For Signal Processing*, pages 13–16. 2019.

[20] L. Pavelková and K. Belda. State estimation and model predictive control for the systems with uniform noise. In *Proc. of 11th IFAC Symposium on Dynamics and Control of Process Systems, including Biosystems (DYCOPS-CAB 2016)*, pages 967–972, 2016.

[21] L. Pavelková and L. Jirsa. Approximate recursive Bayesian estimation of state space model with uniform noise. In *Proceedings of the 15th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, pages 388–394. SCITEPRESS – Science and Technology Publications, Porto, Portugal, 2018.

[22] L. Kuklišová Pavelková, L. Jirsa, and A. Quinn. Bayesian filtering for states uniformly distributed on a parallelotopic support. In *2019 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT) (ISSPIT2019)*, Ajman, United Arab Emirates, dec 2019.

[23] L. Kuklišová Pavelková, L. Jirsa, and A. Quinn. Fully probabilistic design for knowledge fusion between Bayesian filters under uniform disturbances. *Knowledge-Based Systems*, 238:107879, 2022.

[24] Z. Peng, Y. Li, and G. Hao. The Research on Distributed Fusion Estimation Based on Machine Learning. *IEEE Access*, 8:38174–38184, 2020.

[25] J. Tavares R. R. Pinho and M. Correia. Efficient approximation of the Mahalanobis distance for tracking with the Kalman filter. *Computational Modeling of Objects Represented in Images-Fundamentals, Methods and Applications, First International Symposium CompIMAGE 2006, Coimbra, Portugal, October 20-21,2006*, page 349–354, 2006.

[26] L. Ros, A. Sabater i Pruna, and F. Thomas. An ellipsoid calculus based on propagation and fusion. *IEEE Transactions on Systems Man and Cybernetics Part B (Cybernetics)*, 32:430–443, 08 2002.

[27] Q. Shen, J. Liu, X. Zhou, W. Qin, L. Wang, and Q. Wang. Centralized Fusion Methods for Multi-Sensor System With Bounded Disturbances. *IEEE Access*, 7:141612–141626, 2019.

[28] A. Vicino and G. Zappa. Sequential approximation of feasible parameter sets for identification with set membership uncertainty. In *IEEE Transactions on Automatic Control*, volume 41(6), pages 774–785. 1996.

[29] K. Weiss, T. M. Khoshgoftaar, and D. Wang. A Survey of Transfer Learning. *Journal of Big data*, 3(1):1–40, 2016.

[30] Y. Zheng. Methodologies for Cross-Domain Data Fusion: An Overview. *IEEE Transactions on Big Data*, 1(1):16–34, 2015.