entropy

MDPI

Article

Multisensor Estimation Fusion with Gaussian Process for Nonlinear Dynamic Systems

Yiwei Liao [1], Jiangqiong Xie [1], Zhiguo Wang [2,3] and Xiaojing Shen [1,*]

1 School of Mathematics, Sichuan University, Chengdu 610064, China; lyw@stu.scu.edu.cn (Y.L.); xiejiangqiong@163.com (J.X.)
2 School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen 518172, China; wangzhiguo@cuhk.edu.cn
3 Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei 230026, China
* Correspondence: shenxj@scu.edu.cn

Received: 9 October 2019; Accepted: 11 November 2019 ; Published: 16 November 2019

Abstract: The Gaussian process is gaining increasing importance in different areas such as signal processing, machine learning, robotics, control and aerospace and electronic systems, since it can represent unknown system functions by posterior probability. This paper investigates multisensor fusion in the setting of Gaussian process estimation for nonlinear dynamic systems. In order to overcome the difficulty caused by the unknown nonlinear system models, we associate the transition and measurement functions with the Gaussian process regression models, then the advantages of the non-parametric feature of the Gaussian process can be fully extracted for state estimation. Next, based on the Gaussian process filters, we propose two different fusion methods, centralized estimation fusion and distributed estimation fusion, to utilize the multisensor measurement information. Furthermore, the equivalence of the two proposed fusion methods is established by rigorous analysis. Finally, numerical examples for nonlinear target tracking systems demonstrate the equivalence and show that the multisensor estimation fusion performs better than the single sensor. Meanwhile, the proposed fusion methods outperform the convex combination method and the relaxed Chebyshev center covariance intersection fusion algorithm.

Keywords: multisensor estimation fusion; Gaussian process; nonlinear dynamic systems; data driven modeling; target tracking; information fusion

1. Introduction

Estimation in nonlinear systems is extremely important because almost all practical systems involve nonlinearities of one kind or another [1,2], such as target tracking, vehicle navigation, automatic control and robotics [3,4]. In the case of nonlinearities, estimation cannot be obtained analytically in general. Some methods based on exact parametric models and ideas of the Kalman filter (KF) have been developed. The Extended Kalman filter (EKF) [5] was the most common application to nonlinear systems, which simply linearizes all nonlinear functions via Taylor-series expansion and substitutes Jacobian matrices for the linear transformations into the KF equations. The unscented transformation was introduced to address the deficiencies of linearization, namely unscented Kalman filters (UKF) [1], which provided a more direct and explicit mechanism for transforming mean and covariance matrices. Under the Bayesian framework, particle filter (PF) was presented by constructing the posterior probability density function of the state based on all available information in Reference [6]. However, for nonlinear systems, these methods were usually assumed that the nonlinear relationships are known. Lack of modeling accuracy, including the identification of the noise and the model

parameters, was inevitable [7]. In many applications, most real dynamic systems are difficult to obtain the exact models and system noises due to the complexity of the target motion environment, therefore, the parameterized estimation methods may be invalid.

To overcome the limitations of parametric models, researchers have recently employed so called non-parametric methods like the Gaussian process [8] to learn models for dynamic systems. More specifically, the functional representation needs to be learned from the training data before the filtering prediction and update step [9]. Gaussian processes have been increasingly attracting the interests in machine learning, signal processing, robotics, control and aerospace and electronic systems [10–12]. For example, Gaussian process models are used as the surrogate models for complex physics models in Reference [13]. The advantages stemmed from the fact that Gaussian processes take both the noise in the system and the uncertainty in the model into consideration [14]. In the context of modeling the dynamic systems, Gaussian processes can be used as prior over the transition function and measurement function. By analyzing the correlation among given training data, Gaussian process models can provide the posterior distributions over functions through the combination of the prior and the data. For the cases that ground truth data are unavailable or can only be determined approximately, Gaussian process latent variable models were developed in References [15,16] and they were extended to the setting of dynamical robotics systems in Reference [17]. In order to reduce the cubic complexity of Gaussian process training for the fixed number of training points, sparse Gaussian processes were developed (see e.g., [18–23]). K-optimality was used to improve the stability of the Gaussian process prediction in Reference [24].

So far, Gaussian process models have been applied successfully to massive nonlinear dynamic systems. Motivated by modeling human motion, learning nonlinear dynamical models with Gaussian process was investigated in Reference [16]. Many filtering methods, such as the Gaussian process extended Kalman filters (GP-EKF) [25], the Gaussian process unscented Kalman filters (GP-UKF) [26], the Gaussian process particle filters (GP-PF) [27], GP-BayesFilters [14] and the Gaussian process assumed density filters (GP-ADF) [7,10] were derived by incorporating the Gaussian process transition model and Gaussian process measurement model into the classic filters. GP-ADF was an efficient form of assumed density filtering (ADF) introduced in References [28–30] and propagated the full Gaussian density [7]. Although Gaussian processes have been around for decades, they mainly focus on single sensor systems.

With the rapid development of the sensor technology, computer science, communication technology and information technology, multisensor systems are widely used in military and civil fields [31–35]. Benefited from the application of multiple sensors, multisensor data fusion makes more comprehensive and accurate decision by integrating the available information from multiple sensors and has attracted lots of research interests. Generally speaking, the multisensor estimation fusion mainly contains centralized estimation fusion and distributed estimation fusion. Centralized estimation fusion is that a central processor receives all measurement data from sensors without processing and uses them to estimate the state [36,37]. In general, many filtering algorithms in single sensor system can be applied to the multisensor systems, since measurement value can be stacked and regarded as one measurement. On the other hand, distributed estimation fusion has several advantages in terms of reliability, survivability, communication bandwidth and computational burdens [38–42], which make it desirable in real applications such as surveillance, tracking and battle management. In the distributed setting, each local sensor deals with its own measurement data and transmits the local state estimation to the fusion center for the purpose of more accurate estimation fusion. So far, a variety of distributed fusion methods have been investigated for different occasions, such as References [37,40,43–46]. A convex combination method was given to fuse the local estimates in Reference [43]. For the unavailable cross correlation matrix problem, a relaxed Chebyshev center covariance intersection (RCC-CI) algorithm was also provided to fuse the local estimates in Reference [37]. A current review of distributed multisensor data fusion under unknown correlation can be seen in Reference [47]. However, these multisensor estimation fusion methods are mainly based on

the exact dynamic models or known nonlinear functions. For the cases in which accurate parametric models are difficult to obtain, it is worth considering integrating Gaussian processes with multisensor estimation fusion to improve the system's performance.

In this paper, we focus on multisensor estimation fusion including the centralized and distributed fusion methods with Gaussian process for nonlinear dynamic systems. Firstly, given the training data, we learn the dynamic models with the Gaussian process and derive multisensor estimation fusion methods based on the Gaussian process models for nonlinear dynamic systems, which can avoid inappropriate parametric models and improve predictive ability. Combining with the single senor GP-ADF and GP-UKF, respectively, the prediction step and update step of the multisensor estimation fusion are provided. In general, it is hard to analyze the performance of the non-parametric fusion methods. Since the Gaussian process fusion methods have analytic mean and covariance, we show that the distributed estimation fusion is equivalent to the centralized estimation fusion with the single sensor cross terms being full column rank. Numerical examples show that the equivalence is satisfied under the condition and the multisensor estimation fusion performs better than the single sensor. We also compare the proposed fusion methods with the RCC-CI [37] algorithm and the convex combination method [43]. In the Gaussian process model setting, the simulation results show that the multisensor estimation fusion methods outperform the RCC-CI algorithm and the convex combination method, as far as the estimation accuracy is concerned.

This article also extends our earlier work [48]. Compared with the conference paper, the main differences are as follows:

- Detailed proofs are provided.
- The enhancement of the multisensor fusion algorithm with GP-UKF is presented.
- Additional set of extensive experiments are carried out.
- The equivalence condition of Proposition 1 is analyzed.
- Comparison between GP-ADF fusion and GP-UKF is given.

The rest of the paper is organized as follows. In Section 2, the problem formulation and the Gaussian process are introduced. In Section 3, the centralized estimation fusion and the distributed estimation fusion methods are presented. In Section 4, simulations are provided to confirm our analysis. Some conclusions are drawn in Section 5.

2. Preliminaries

2.1. Problem Formulation

In this paper, we consider the state estimation problem of a nonlinear dynamic system with additive noise and N ($N \geq 2$) sensor measurements. The multisensor nonlinear dynamic system with state equation and N measurement equations (see Figure 1) is described as follows:

$$\mathbf{x}_k = h(\mathbf{x}_{k-1}) + \mathbf{w}_k, \quad k = 0, 1, \ldots, \tag{1}$$

$$\mathbf{y}_k^m = g^m(\mathbf{x}_k) + \mathbf{v}_k^m, \quad m = 1, \ldots, N, \tag{2}$$

where $\mathbf{x}_k \in \mathcal{R}^D$ is the state of the dynamic system at time k and $\mathbf{y}_k^m \in \mathcal{R}^{l_m}$ is the measurement of the m_{th} sensor at time k, $m = 1, \ldots, N$. $h(\mathbf{x}_k)$ is the transition function of the state $\mathbf{x}_k$ and $g^m(\mathbf{x}_k)$ is the nonlinear measurement function of $\mathbf{x}_k$ at the m_{th} sensor. $\mathbf{w}_k \sim N(\mathbf{0}, \mathbf{Q}_k)$ is a Gaussian system noise and $\mathbf{v}_k^m \sim N(\mathbf{0}, \mathbf{R}_k^m)$ is a Gaussian measurement noise, which is independent of each other.

Our goal is to estimate the state from all the available sensor measurement information. Gaussian processes are regarded as the prior of the transition function $h(\mathbf{x}_{k-1})$ and the sensor measurement function $g^m(\mathbf{x}_k)$, $m = 1, \ldots, N$, respectively. Then we make inference about the posterior distribution of the transition function and the sensor measurement function. The Gaussian process model represents a powerful tool to make Bayesian inference about functions [49].

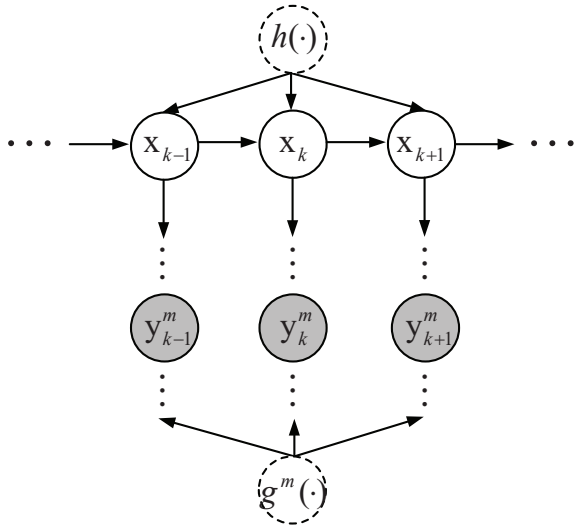

Figure 1. Graphical structure for multisensor nonlinear dynamic systems.

2.2. Gaussian Processes

A Gaussian process is defined over functions, which is a generalization of the Gaussian probability distribution [8]. It means that we should consider inference directly in function space. Also, it gives a formal definition, "the Gaussian process is defined a collection of random variables, any finite number of which have a joint Gaussian distribution," in [8].

Similar to a Gaussian distribution, knowledge of both mean and covariance function can specify a Gaussian process. The mean function $m(\mathbf{x})$ and covariance function (also called a kernel) $k(\mathbf{x}, \mathbf{x}')$ of a Gaussian process $f(\mathbf{x})$ are defined as follows,

$$\begin{aligned} m(\mathbf{x}) &= \mathbb{E}[f(\mathbf{x})], \\ k(\mathbf{x}, \mathbf{x}') &= \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))], \end{aligned}$$

where $\mathbb{E}[\cdot]$ denotes the expectation. Thus, we write the Gaussian process as

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')).$$

Unless stated otherwise, a prior mean function is usually assumed to be 0. The choice of the covariance functions depends on the application [14]. In this paper, we employ the most commonly-used squared exponential (SE) kernel in machine learning, which is the prototypical stationary covariance function and useful for modeling particularly smooth functions. It is defined as

$$\begin{aligned} \mathrm{Cov}(f(\mathbf{x}), f(\mathbf{x}')) &= k(\mathbf{x}, \mathbf{x}') \\ &= \alpha^2 \exp\{-\frac{1}{2}(\mathbf{x} - \mathbf{x}')^{\mathrm{T}} \mathbf{\Lambda}^{-1} (\mathbf{x} - \mathbf{x}')\}, \end{aligned} \tag{3}$$

where parameter α^2 represents the variance of the function f that controls the uncertainty of predictions in areas of less training sample density and parameter $\mathbf{\Lambda}$ is a diagonal matrix of the characteristic length-scales of the SE kernel. Other commonly employed kernel functions can be seen in Reference [8].

A Gaussian process implies a distribution over functions based upon the obtained training data. Assume that we have obtained a set of training data $\mathcal{X} = \{\mathbf{X}, \mathbf{y}\}$. $\mathbf{X}$ and $\mathbf{y}$ are made up of multiple samples drawn from the following standard Gaussian process regression model

$$y = f(\mathbf{x}) + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \sigma_\varepsilon^2), \tag{4}$$

where $f : \mathcal{R}^D \to \mathcal{R}$ and $f(\mathbf{x}) \sim \mathcal{GP}$, $\mathcal{N}$ denotes the normalized Gaussian probability density function. Note that Gaussian process regression uses the fact that any finite set of training data and testing data of a Gaussian process are jointly Gaussian distributed. Let $\boldsymbol{\theta} = \{\alpha^2, \boldsymbol{\Lambda}, \sigma_\varepsilon^2\}$, called the hyper-parameters of the Gaussian process. Using the evidence maximization method [8,50], we obtain a point estimate

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \log p(\mathbf{y} \mid \mathbf{X}, \boldsymbol{\theta}),$$

from the known training data [51]. We can solve the optimization problem with numerical optimization techniques such as conjugate gradient ascent [8,14]. Next, we infer the posterior distribution over function value $f(\mathbf{x}_*)$ from training data for each inputs $\mathbf{x}_* \in \mathcal{R}^D$. In addition, the test input $\mathbf{x}_*$ can be categorized into two classes, relying on whether it is uncertain or not. The corresponding predictive distribution over $f(\mathbf{x}_*)$ will be presented as follows.

(1) **Predictive Distribution over Univariate Function**

Suppose that the training data is $\mathcal{X} = (\mathbf{X}, \mathbf{y})$=$\{(\mathbf{x}_i, y_i) : i = 1, \ldots, n\}$, where $\mathbf{x}_i$ is a D-dimensional input vector, y_i is a scalar output, and n is the number of training data. Next, two cases, deterministic inputs and uncertain inputs, are considered.

- **Deterministic Inputs**

Conditioned on the training data and the deterministic test input $\mathbf{x}_*$, the predictive distribution over $f(\mathbf{x}_*)$ is a Gaussian with the mean

$$m_f(\mathbf{x}_*) = \mathbb{E}[f_*] = \mathbf{k}_*^{\mathrm{T}}(\mathbf{K} + \sigma_\varepsilon^2 \mathbf{I})^{-1} y = \mathbf{k}_*^{\mathrm{T}} \boldsymbol{\beta}, \tag{5}$$

and the variance

$$\sigma_f^2(\mathbf{x}_*) = \mathrm{Var}_f[f_*] = k_{**} - \mathbf{k}_*^{\mathrm{T}}(\mathbf{K} + \sigma_\varepsilon^2 \mathbf{I})^{-1} \mathbf{k}_*, \tag{6}$$

where $\mathrm{Var}_f[\cdot]$ represents the variance with respect to f,

$$\begin{aligned} \mathbf{k}_* &:= k(\mathbf{X}, \mathbf{x}_*), \\ k_{**} &:= k(\mathbf{x}_*, \mathbf{x}_*), \\ \boldsymbol{\beta} &:= (\mathbf{K} + \sigma_\varepsilon^2 \mathbf{I})^{-1} \mathbf{y}, \end{aligned}$$

and $\mathbf{K}$ is the kernel matrix with each element satisfying $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. The uncertainty of prediction is characterized by the variance $\sigma_f^2(\mathbf{x}_*)$. More details can be seen in Reference [8].

- **Uncertain Inputs**

When the test input is uncertain, namely $\mathbf{x}_*$ has a probability distribution, the prediction problem is relatively difficult. Let us consider the predictive distribution of $f(\mathbf{x}_*)$ for the uncertain input $\mathbf{x}_* \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. According to the results in Reference [7] (or reviewing References [52,53]), we introduce the predictive distribution over the function value as

$$p(f(\mathbf{x}_*)|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \int p(f(\mathbf{x}_*)|\mathbf{x}_*) p(\mathbf{x}_*|\boldsymbol{\mu}, \boldsymbol{\Sigma}) d\mathbf{x}_*, \tag{7}$$

where the mean and variance of the distribution $p(f(\mathbf{x}_*)|\mathbf{x}_*)$ are provided with Equations (5) and (6), respectively. Based on the conditional expectation and variance formulae, it yields the mean μ_* and variance σ_*^2 of the distribution $p(f(\mathbf{x}_*)|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ in closed form. In particular, we have

$$\begin{aligned}\mu_* &= \mathbb{E}_{\mathbf{x}_*}[\mathbb{E}_f(f(\mathbf{x}_*)|\mathbf{x}_*)|\boldsymbol{\mu}, \boldsymbol{\Sigma}] \\ &= \mathbb{E}_{\mathbf{x}_*}[m_f(\mathbf{x}_*)|\boldsymbol{\mu}, \boldsymbol{\Sigma}] \\ &= \int m_f(\mathbf{x}_*)\mathcal{N}(\mathbf{x}_*|\boldsymbol{\mu}, \boldsymbol{\Sigma})d\mathbf{x}_* \\ &= \boldsymbol{\beta}^{\mathrm{T}}\mathbf{l}, \end{aligned} \tag{8}$$

and

$$\begin{aligned}\sigma_*^2 &= \mathbb{E}_{\mathbf{x}_*}[m_f(\mathbf{x}_*)^2|\boldsymbol{\mu}, \boldsymbol{\Sigma}] + \mathbb{E}_{\mathbf{x}_*}[\sigma_f^2(\mathbf{x}_*)|\boldsymbol{\mu}, \boldsymbol{\Sigma}] \\ &\quad - \mathbb{E}_{\mathbf{x}_*}[m_f(\mathbf{x}_*)|\boldsymbol{\mu}, \boldsymbol{\Sigma}]^2 \\ &= \boldsymbol{\beta}^{\mathrm{T}}\tilde{\mathbf{L}}\boldsymbol{\beta} + \alpha^2 - tr((\mathbf{K} + \sigma_\varepsilon^2\mathbf{I})^{-1}\tilde{\mathbf{L}}) - (\mu_*)^2. \end{aligned} \tag{9}$$

Note that $\mathbf{l}$ in Equation (8) is from $\mathbf{l} = [l_1, \ldots, l_n]^{\mathrm{T}}$,

$$\begin{aligned}l_j = \int k_f(\mathbf{x}_j, \mathbf{x}_*)p(\mathbf{x}_*)d\mathbf{x}_* &= \alpha^2|\boldsymbol{\Sigma}\boldsymbol{\Lambda}^{-1} + \mathbf{I}|^{-\frac{1}{2}} \\ &\times \exp\{-\frac{1}{2}(\mathbf{x}_j - \boldsymbol{\mu})^{\mathrm{T}}(\boldsymbol{\Sigma} + \boldsymbol{\Lambda})^{-1}(\mathbf{x}_j - \boldsymbol{\mu})\}.\end{aligned}$$

$tr(\cdot)$ represents the trace in Equation (9) and we have

$$\begin{aligned}\tilde{L}_{ij} = &\frac{k_f(\mathbf{x}_i, \boldsymbol{\mu})k_f(\mathbf{x}_j, \boldsymbol{\mu})}{|2\boldsymbol{\Sigma}\boldsymbol{\Lambda}^{-1} + \mathbf{I}|^{\frac{1}{2}}} \\ &\times \exp\{(\tilde{\mathbf{z}}_{ij} - \boldsymbol{\mu})^{\mathrm{T}}(\boldsymbol{\Sigma} + \frac{1}{2}\boldsymbol{\Lambda})^{-1}\boldsymbol{\Sigma}\boldsymbol{\Lambda}^{-1}(\tilde{\mathbf{z}}_{ij} - \boldsymbol{\mu})\},\end{aligned}$$

and

$$\tilde{\mathbf{z}}_{ij} := \frac{1}{2}(\mathbf{x}_i + \mathbf{x}_j).$$

In general, the predictive distribution in Equation (7) cannot be analytically calculated since it leads to a non-Gaussian distribution, if a Gaussian distribution is mapped through a nonlinear function. Using moment-matching method, the distribution $p(f(\mathbf{x}_*)|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ can be approximated by the Gaussian distribution $\mathcal{N}(\mu_*, \sigma_*^2)$.

(2) **Predictive Distribution over Multivariate Function**

For the model (4), let us turn to the multiple output case that $f : \mathcal{R}^D \to \mathcal{R}^E$, $f \sim \mathcal{GP}$. Following the results in References [7,51], we need to train E Gaussian process regression models independently. The ath model is learned from the training data $[X, \mathbf{y}^a]$, $\mathbf{y}^a = [y_1^a, \ldots, y_n^a]^{\mathrm{T}}$, $a = 1, \ldots, E$, where y_j^a is the ath element of y_j. It implies that any two targeted dimensions are conditionally independent given the input.

For any deterministic input $\mathbf{x}_*$, the mean and variance of each target dimension are obtained by Equations (5) and (6). The predictive mean of $f(\mathbf{x}_*)$ is a vector stacked by E targeted dimension mean, and the corresponding covariance is a diagonal matrix whose diagonal element is the variance of each targeted dimension.

With the uncertain input $\mathbf{x}_* \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, the predictive mean $\boldsymbol{\mu}_*$ of $f(\mathbf{x}_*)$ also is the stacked E predictive mean $\boldsymbol{\mu}_*^a$ given by Equation (8), $a = 1, \ldots, E$. Unlike predicting at deterministic

inputs, targeted dimensions are dependent due to the uncertain input. Denote $f_a^* = f_a(\mathbf{x}_*)$ and the corresponding predictive covariance matrix is given by

$$\begin{aligned}&\boldsymbol{\Sigma}_* \mid \boldsymbol{\mu}, \boldsymbol{\Sigma} \\ &= \begin{bmatrix} \mathrm{Var}[f_1^* | \boldsymbol{\mu}, \boldsymbol{\Sigma}] & \dots & \mathrm{Cov}[f_1^*, f_E^* | \boldsymbol{\mu}, \boldsymbol{\Sigma}] \\ \vdots & \ddots & \vdots \\ \mathrm{Cov}[f_E^*, f_1^* | \boldsymbol{\mu}, \boldsymbol{\Sigma}] & \dots & \mathrm{Var}[f_E^* | \boldsymbol{\mu}, \boldsymbol{\Sigma}] \end{bmatrix}. \end{aligned} \tag{10}$$

It is obvious that the predictive covariance is no longer a diagonal matrix. Using Equation (9), we can obtain the diagonal element of covariance matrix (10). For each $a, b \in \{1, \dots, E\}$, the off-diagonal elements satisfy

$$\mathrm{Cov}[f_a^*, f_b^* | \boldsymbol{\mu}, \boldsymbol{\Sigma}] = \mathbb{E}_{f, \mathbf{x}_*}[f_a(\mathbf{x}_*) f_b(\mathbf{x}_*) | \boldsymbol{\mu}, \boldsymbol{\Sigma}] - \mu_*^a \mu_*^b.$$

Given $\mathbf{x}_*$, $f_a(\mathbf{x}_*)$ and $f_b(\mathbf{x}_*)$ are independent, then we have

$$\begin{aligned}&\mathbb{E}_{f, \mathbf{x}_*}[f_a^* f_b^* | \boldsymbol{\mu}, \boldsymbol{\Sigma}] \\ &= \int \int f_a^* f_b^* p(f_a, f_b | \mathbf{x}_*) p(\mathbf{x}_* | \boldsymbol{\mu}, \boldsymbol{\Sigma}) df d\mathbf{x}_* \\ &= \boldsymbol{\beta}_a^{\mathrm{T}} \int k_f^a(\mathbf{X}, \mathbf{x}_*) k_f^b(\mathbf{x}_*, \mathbf{X}) p(\mathbf{x}_* | \boldsymbol{\mu}, \boldsymbol{\Sigma}) d\mathbf{x}_* \boldsymbol{\beta}_b, \end{aligned} \tag{11}$$

where $\boldsymbol{\beta}_a, \boldsymbol{\beta}_b$ can be similarly obtained in Equation (5). For notational simplicity, define

$$\mathbf{L} := \int k_f^a(\mathbf{X}, \mathbf{x}_*) k_f^b(\mathbf{x}_*, \mathbf{X}) p(\mathbf{x}_* | \boldsymbol{\mu}, \boldsymbol{\Sigma}) d\mathbf{x}_*,$$

where

$$\begin{aligned} L_{ij} &= \alpha_a^2 \alpha_b^2 |(\boldsymbol{\Lambda}_a^{-1} + \boldsymbol{\Lambda}_b^{-1}) \boldsymbol{\Sigma} + \mathbf{I}|^{-\frac{1}{2}} \\ &\quad \times \exp\{-\frac{1}{2}(\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}} (\boldsymbol{\Lambda}_a + \boldsymbol{\Lambda}_b)^{-1} (\mathbf{x}_i - \mathbf{x}_j)\} \\ &\quad \times \exp\{-\frac{1}{2}(\mathbf{z}_{ij} - \boldsymbol{\mu})^{\mathrm{T}} \mathbf{R}^{-1} (\mathbf{z}_{ij} - \boldsymbol{\mu})\}, \\ \mathbf{R} &:= (\boldsymbol{\Lambda}_a^{-1} + \boldsymbol{\Lambda}_b^{-1})^{-1} + \boldsymbol{\Sigma}, \\ \mathbf{z}_{ij} &:= \boldsymbol{\Lambda}_b (\boldsymbol{\Lambda}_a + \boldsymbol{\Lambda}_b)^{-1} \mathbf{x}_i + \boldsymbol{\Lambda}_a (\boldsymbol{\Lambda}_a + \boldsymbol{\Lambda}_b)^{-1} \mathbf{x}_j. \end{aligned}$$

Thus, we also approximate the distribution $p(f(\mathbf{x}_*) | \boldsymbol{\mu}, \boldsymbol{\Sigma})$ with the Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*)$. More details can be found in Reference [51].

3. Multisensor Estimation Fusion

In this section, an essential prerequisite is that the transition function (1) and the measurement functions (2) are either not known or no longer accessible. Thus, we model the latent functions with Gaussian processes. For the N-sensor dynamic systems (1) and (2), suppose that we have obtained some training data of the target state and sensor measurement. In the following, we discuss the centralized and distributed estimation fusion methods to estimate the state from a sequence of noisy sensor measurements.

3.1. Centralized Estimation Fusion

First, we consider the centralized estimation fusion method. To facilitate fusion, we stack the multisensor measurement information as follows

$$\begin{aligned} \mathbf{y}_k &= [(\mathbf{y}_k^1)^{\mathrm{T}}, \ldots, (\mathbf{y}_k^N)^{\mathrm{T}}]^{\mathrm{T}}, \\ g(\mathbf{x}_k) &= [(g^1(\mathbf{x}_k))^{\mathrm{T}}, \ldots, (g^N(\mathbf{x}_k))^{\mathrm{T}}]^{\mathrm{T}}, \\ \mathbf{v}_k &= [(\mathbf{v}_k^1)^{\mathrm{T}}, \ldots, (\mathbf{v}_k^N)^{\mathrm{T}}]^{\mathrm{T}}. \end{aligned} \tag{12}$$

Thus the dynamic system (1) and (2) can be rewritten as

$$\mathbf{x}_k = h(\mathbf{x}_{k-1}) + \mathbf{w}_k, \tag{13}$$

$$\mathbf{y}_k = g(\mathbf{x}_k) + \mathbf{v}_k, \tag{14}$$

where the covariance of the noise $\mathbf{v}_k$ is given by

$$\begin{aligned} \mathrm{Cov}(\mathbf{v}_k) =& \mathbf{\Sigma}_k = diag(\mathbf{\Sigma}_k^1, \ldots, \mathbf{\Sigma}_k^N), \\ \mathrm{Cov}(\mathbf{v}_k^m) =& \mathbf{\Sigma}_k^m, \quad m = 1, \ldots, N. \end{aligned}$$

Considering the Gaussian process dynamic system setup, two Gaussian process models can be trained using evidence maximization. $\mathcal{GP}_h$ models the mapping $\mathbf{x}_{k-1} \mapsto \mathbf{x}_k$, $\mathcal{R}^D \to \mathcal{R}^D$ and $\mathcal{GP}_g$ models the mapping $\mathbf{x}_k \mapsto \mathbf{y}_k$, $\mathcal{R}^D \to \mathcal{R}^{N*E}$.

As we know, the key of the estimation is to recursively infer the posterior distribution over the state $\mathbf{x}_k$ of the dynamic system (13)–(14), which are based on all the sensor measurements $\mathbf{y}_{1:k} = \{\mathbf{y}_\tau\}_{\tau=1}^k$, including the prediction step and the update step. In particular, the prediction step uses the posterior distribution from the previous time step to produce a predictive distribution of the state at the current time step. In the update step, the current predictive distribution is combined with current measurement information to refine the posterior distribution at the current time step. Next, centralized estimation fusion methods with GP-ADF and GP-UKF are presented, respectively.

(1) **Fusion with GP-ADF**

Note that some approximation methods are required to obtain an analytic posterior distribution for the nonlinear dynamic systems. In particular, for the Gaussian process dynamic system, it is easy to make some Gaussian approximation to the posterior distribution. GP-ADF exploits the fact that the true moments of the Gaussian process predictive distribution can be computed in closed form. The predictive distribution is approximated by a Gaussian with the exact predictive mean and covariance [7]. Based on the following lemma, we present the centralized estimation fusion method with GP-ADF.

Lemma 1. *For the Gaussian process dynamic system (13)–(14), the posterior distribution of the state $\mathbf{x}_k$ can be approximated by the Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_k^e, \mathbf{C}_k^e)$, in which the mean $\boldsymbol{\mu}_k^e$ and covariance $\mathbf{C}_k^e$ are given as [7]:*

$$\boldsymbol{\mu}_k^e = \boldsymbol{\mu}_k^p + \mathbf{C}_{\mathbf{x}_k\mathbf{y}_k}(\mathbf{C}_k^y)^{-1}(\mathbf{y}_k - \boldsymbol{\mu}_k^y), \tag{15}$$

$$\mathbf{C}_k^e = \mathbf{C}_k^p - \mathbf{C}_{\mathbf{x}_k\mathbf{y}_k}(\mathbf{C}_k^y)^{-1}\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k}^{\mathrm{T}}, \tag{16}$$

where

$$\begin{aligned}
\boldsymbol{\mu}_k^p &= \mathbb{E}[\mathbf{x}_k|\mathbf{y}_{1:k-1}], \\
\mathbf{C}_k^p &= \mathrm{Cov}(\mathbf{x}_k|\mathbf{y}_{1:k-1}), \\
\boldsymbol{\mu}_k^y &= \mathbb{E}[\mathbf{y}_k|\mathbf{y}_{1:k-1}], \\
\mathbf{C}_k^y &= \mathrm{Cov}(\mathbf{y}_k|\mathbf{y}_{1:k-1}), \\
\mathbf{C}_{x_k y_k} &= \mathrm{Cov}(\mathbf{x}_k, \mathbf{y}_k|\mathbf{y}_{1:k-1}).
\end{aligned} \tag{17}$$

Remark 1. *Since GP-ADF algorithm [7] is a state estimation method in the single sensor Gaussian process dynamic system, it is applicable to the multisensor system with all the sensor measurements stacked into a vector. Therefore, Equation* (15) *and Equation* (16) *are called the centralized estimation fusion method with GP-ADF here.*

Then, let us present the prediction step and update step of the above fusion method in detail.

- **Prediction Step**

First, this step needs to use the previous posterior distribution $p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1}) \approx \mathcal{N}(\boldsymbol{\mu}_{k-1}^e, \mathbf{C}_{k-1}^e)$ to produce a current predictive distribution $p(\mathbf{x}_k|\mathbf{y}_{1:k-1})$. We write the predictive distribution as

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

and $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ is a Gaussian distribution due to $h \sim \mathcal{GP}$ and $\mathbf{w}_k \sim \mathcal{N}(0, \boldsymbol{\Sigma}_{\mathbf{w}_k})$. From this, an analogy with Equation (7) yields the mean $\boldsymbol{\mu}_k^p$ and the variance $\mathbf{C}_k^p$ of the state $\mathbf{x}_k$ conditioned on the measurement $\mathbf{y}_{1:k-1}$. In particular,

$$\begin{aligned}
\boldsymbol{\mu}_k^p &= \mathbb{E}(\mathbf{x}_k|\mathbf{y}_{1:k-1}) \\
&= \mathbb{E}[h(\mathbf{x}_{k-1})|\mathbf{y}_{1:k-1}] \\
&= \mathbb{E}_{\mathbf{x}_{k-1}}[\mathbb{E}_h(h(\mathbf{x}_{k-1})|\mathbf{x}_{k-1})|\mathbf{y}_{1:k-1}].
\end{aligned} \tag{18}$$

$\boldsymbol{\mu}_k^p$ is a D-dimension mean vector and the computation of every target dimension can be seen in Equation (8). The corresponding covariance matrix

$$\begin{aligned}
\mathbf{C}_k^p &= \mathrm{Cov}(h(\mathbf{x}_{k-1})|\mathbf{y}_{1:k-1}) + \mathrm{Cov}(\mathbf{w}_k) \\
&= \boldsymbol{\Sigma}_h + \boldsymbol{\Sigma}_{w_k},
\end{aligned} \tag{19}$$

where $\boldsymbol{\Sigma}_{w_k}$ is the covariance matrix of transition noise obtained by the evidence maximization method, and the computation of $\boldsymbol{\Sigma}_h$ can be seen in Equation (10).

- **Update Step**

Next, we approximate the joint distribution $p(\mathbf{x}_k, \mathbf{y}_k|\mathbf{y}_{1:k-1})$ with a joint Gaussian $\mathcal{N}(\boldsymbol{\mu}_k, \mathbf{C}_k)$, where

$$\boldsymbol{\mu}_k = \begin{bmatrix} \boldsymbol{\mu}_k^p \\ \boldsymbol{\mu}_k^y \end{bmatrix}, \ \mathbf{C}_k = \begin{bmatrix} \mathbf{C}_k^p & \mathbf{C}_{\mathbf{x}_k\mathbf{y}_k} \\ \mathbf{C}_{\mathbf{x}_k\mathbf{y}_k}^{\mathrm{T}} & \mathbf{C}_k^y \end{bmatrix}.$$

Note that we are not aiming at approximating the distribution $p(\mathbf{x}_k|\mathbf{y}_{1:k})$ directly. To obtain the joint approximation Gaussian distribution, we use the Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_k^y, \mathbf{C}_k^y)$ to approximate the distribution $p(\mathbf{y}_k|\mathbf{y}_{1:k-1})$, which can be done in a same way to the prediction step due to

$$p(\mathbf{y}_k|\mathbf{y}_{1:k-1}) = \int p(\mathbf{y}_k|\mathbf{x}_k) p(\mathbf{x}_k|\mathbf{y}_{1:k-1}) d\mathbf{x}_k.$$

On the other hand, the cross term satisfies

$$\mathbf{C}_{\mathbf{x}_k,\mathbf{y}_k} = \mathbb{E}_{\mathbf{x}_k,g}[\mathbf{x}_k(\mathbf{y}_k)^{\mathrm{T}}|\mathbf{y}_{1:k-1}] - \boldsymbol{\mu}_k^p(\boldsymbol{\mu}_k^y)^{\mathrm{T}}.$$

For the unknown term $\mathbb{E}_{\mathbf{x}_k,g}[\mathbf{x}_k(\mathbf{y}_k)^{\mathrm{T}}|\mathbf{y}_{1:k-1}]$, we have

$$\begin{aligned}
&\mathbb{E}_{\mathbf{x}_k,g_{\tilde{a}}}[\mathbf{x}_k\mathbf{y}_k^{\tilde{a}}|\mathbf{y}_{1:k-1}] = \mathbb{E}_{\mathbf{x}_k,g_{\tilde{a}}}[\mathbf{x}_k g_{\tilde{a}}(\mathbf{x}_k)|\mathbf{y}_{1:k-1}] \\
&= \int \mathbf{x}_k \left[\sum_{i=1}^{n} \beta_i^{\tilde{a}} k_{g^{\tilde{a}}}(\mathbf{x}_k,\mathbf{x}_i)\right] p(\mathbf{x}_k)d\mathbf{x}_k \\
&= \sum_{i=1}^{n} \beta_i^{\tilde{a}} \int \mathbf{x}_k c_1 \mathcal{N}(\mathbf{x}_k|\mathbf{x}_i,\Lambda_{\tilde{a}})\mathcal{N}(\mathbf{x}_k|\boldsymbol{\mu}_k^p,\mathbf{C}_k^p)d\mathbf{x}_k \\
&= c_1(c_2)^{-1} \sum_{i=1}^{n} \beta_i^{\tilde{a}} \psi(x_i,\boldsymbol{\mu}_k^p),
\end{aligned} \tag{20}$$

where $\tilde{a} = 1, 2, \ldots, N * E$, c_1^{-1} is the normalization constant of the unnormalized SE kernel, $\psi(\mathbf{x}_i, \boldsymbol{\mu}_k^p)$ is the mean of a new Gaussian distribution that is the product of two Gaussian density functions and c_2^{-1} is the normalization constant of the new Gaussian distribution. Consequently, from the joint distribution $p(\mathbf{x}_k, \mathbf{y}_k|\mathbf{y}_{1:k-1})$, we obtain the posterior distribution $p(\mathbf{x}_k|\mathbf{y}_{1:k}) \approx \mathcal{N}(\boldsymbol{\mu}_k^e, \mathbf{C}_k^e)$ with the mean and variance as Equations (15) and (16), respectively.

Remark 2. *From Equations* (15) *and* (16)*, we can see that the centralized fusion method with GP-ADF is similar to the unified optimal linear estimation fusion [40,54]. The difference is that the computational way of* $\boldsymbol{\mu}_k^p, \mathbf{C}_k^p, \boldsymbol{\mu}_k^y, \mathbf{C}_k^y$ *and* $\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k}$ *and they are mainly based on the training data due to the nonparametric dynamic system model.*

(2) **Fusion with GP-UKF**

Following the single sensor GP-UKF (see e.g., References [14,26]), we present the prediction step and update step of the GP-UKF fusion.

- **Prediction Step**

At time $k-1$, based on the unscented transform [1], we obtain $\mathcal{X}_{k-1}$ containing $2D+1$ points through the mean $\boldsymbol{\mu}_{k-1}^e$ and covariance $\mathbf{C}_{k-1}^e$. Using the GP prediction model, the transformed set is given by

$$\bar{\mathcal{X}}_k^{[i]} = \mathcal{GP}_h^{[\mu]}(\mathcal{X}_{k-1}^{[i]}),\ for\ i = 1, \ldots, 2D+1, \tag{21}$$

and the process noise is computed as

$$\mathbf{Q}_k = \mathcal{GP}_h^{[\Sigma]}(\boldsymbol{\mu}_{k-1}^e), \tag{22}$$

where $\mathcal{GP}_h^{[\mu]}$ and $\mathcal{GP}_h^{[\Sigma]}$ are the mean and covariance models with respect to the Gaussian process $\mathcal{GP}_h$. The predictive mean and covariance are

$$\boldsymbol{\mu}_k^p = \sum_{i=1}^{2D+1} \mathbf{W}^{[i]} \bar{\mathcal{X}}_k^{[i]}, \tag{23}$$

$$\mathbf{C}_k^p = \sum_{i=1}^{2D+1} \mathbf{W}^{[i]} (\bar{\mathcal{X}}_k^{[i]} - \boldsymbol{\mu}_k^p)(\bar{\mathcal{X}}_k^{[i]} - \boldsymbol{\mu}_k^p)^{\mathrm{T}} + \mathbf{Q}_k, \tag{24}$$

where $\mathbf{W}^{[i]}$ is the weight generated in the unscented transform. Using the predictive mean $\boldsymbol{\mu}_k^p$ and covariance $\mathbf{C}_k^p$, we obtain $\hat{\mathcal{X}}_k^{[i]}$ through the unscented transform. Thus, based on the GP observation model,

$$\hat{\mathcal{Y}}_k^{[i]} = \mathcal{GP}_g^{[\mu]}(\hat{\mathcal{X}}_k^{[i]}) \; for \; i = 1, \ldots, 2D+1, \tag{25}$$

and the observation noise matrix is determined by

$$\mathbf{R}_k = \mathcal{GP}_g^{[\Sigma]}(\hat{\mathcal{X}}_k^{[i]}). \tag{26}$$

Then the predicted observation and innovation covariance are calculated by

$$\hat{\mathbf{y}}_k = \sum_{i=1}^{2D+1} \mathbf{W}^{[i]} \hat{\mathcal{Y}}_k^{[i]}, \tag{27}$$

$$\mathbf{S}_k = \sum_{i=1}^{2D+1} \mathbf{W}^{[i]} (\hat{\mathcal{Y}}_k^{[i]} - \hat{\mathbf{y}}_k)(\hat{\mathcal{Y}}_k^{[i]} - \hat{\mathbf{y}}_k)^{\mathrm{T}} + \mathbf{R}_k. \tag{28}$$

Meanwhile, the cross covariance is given by

$$\mathbf{C}_{\mathbf{x}_k \mathbf{y}_k} = \sum_{i=1}^{2D+1} \mathbf{W}^{[i]} (\hat{\mathcal{X}}_k^{[i]} - \boldsymbol{\mu}_k^p)(\hat{\mathcal{Y}}_k^{[i]} - \hat{\mathbf{y}}_k)^{\mathrm{T}} \tag{29}$$

- **Update Step**

  Finally, the update can be obtained by the following equations:

$$\boldsymbol{\mu}_k^e = \boldsymbol{\mu}_k^p + \mathbf{C}_{\mathbf{x}_k \mathbf{y}_k} \mathbf{S}_k^{-1} (\mathbf{y}_k - \hat{\mathbf{y}}_k), \tag{30}$$

$$\mathbf{C}_k^e = \mathbf{C}_k^p - \mathbf{C}_{\mathbf{x}_k \mathbf{y}_k} \mathbf{S}_k^{-1} \mathbf{C}_{\mathbf{x}_k \mathbf{y}_k}^{\mathrm{T}}. \tag{31}$$

Remark 3. *Note that GP-ADF propagates the full Gaussian density by exploiting specific properties of Gaussian process models [7]. GP-UKF maps the sigma points through the Gaussian process models instead of the parametric functions and the distributions are described by finite-sample approximations. In contrast to GP-UKF, GP-ADF is consistent and moment preserving [7]. In addition GP-EKF requires linearization of the nonlinear prediction and observation models in order to propagate the state and observation, respectively [14]. GP-PF needs to perform one Gaussian process mean and variance computation per particle and has very high computational cost. For more details about these single sensor filtering methods, it can be seen in [7,14], and so forth. Due to the linearization loss of GP-EKF and high computational cost of GP-PF, we do not consider them and only introduce the multisensor estimation fusion with GP-ADF and GP-UKF in this paper.*

3.2. Distributed Estimation Fusion

In this subsection, we mainly present the distributed estimation fusion with GP-ADF. For the dynamic systems (1) and (2), assume that there is a prior on initial state $\mathbf{x}_0$ and each local sensor sends its estimation to the fusion center. Thus, the posterior distribution of m_{th} local state estimation is a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_k^{em}, \mathbf{C}_k^{em})$, where $\boldsymbol{\mu}_k^{em}$ and $\mathbf{C}_k^{em}$ are calculated as follows

$$\boldsymbol{\mu}_k^{em} = \boldsymbol{\mu}_k^{pm} + \mathbf{C}_{\mathbf{x}_k \mathbf{y}_k^m} (\mathbf{C}_k^{ym})^{-1} (\mathbf{y}_k^m - \boldsymbol{\mu}_k^{ym}), \tag{32}$$

$$\mathbf{C}_k^{em} = \mathbf{C}_k^{pm} - \mathbf{C}_{\mathbf{x}_k \mathbf{y}_k^m} (\mathbf{C}_k^{ym})^{-1} \mathbf{C}_{\mathbf{x}_k \mathbf{y}_k^m}^{\mathrm{T}}, \tag{33}$$

with

$$\begin{aligned}
\boldsymbol{\mu}_k^{pm} &= \mathbb{E}[\mathbf{x}_k|\mathbf{y}_{1:k-1}^m], \\
\mathbf{C}_k^{pm} &= \mathrm{Cov}(\mathbf{x}_k|\mathbf{y}_{1:k-1}^m), \\
\boldsymbol{\mu}_k^{ym} &= \mathbb{E}[\mathbf{y}_k^m|\mathbf{y}_{1:k-1}^m], \\
\mathbf{C}_k^{ym} &= \mathrm{Cov}(\mathbf{y}_k^m|\mathbf{y}_{1:k-1}^m), \\
\mathbf{C}_{x_k y_k^m} &= \mathrm{Cov}(\mathbf{x}_k, \mathbf{y}_k^m|\mathbf{y}_{1:k-1}^m).
\end{aligned} \tag{34}$$

Then, the local posterior mean and covariance are transmitted to the fusion center to yield the posterior distribution of global state estimation.

In general, the centralized estimation fusion method with directly fusing raw data has better fusion performance than the distributed estimation fusion method based on the processed data. However, in some cases, the distributed estimation fusion is equivalent to the centralized estimation fusion. In particular, in what follows, we propose a distributed estimation fusion method and prove that the distributed estimation fusion method is equivalent to the above centralized estimation fusion method when all cross terms $\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k^m}$, $m = 1, \ldots, N$, have full column rank.

Proposition 1. *Assume that all cross terms* $\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k^m}$, $m = 1, \ldots, N$ *are full column rank, the distributed estimation fusion is equivalent to the centralized estimation fusion as follows*

$$\begin{aligned}
\boldsymbol{\mu}_k^e = \boldsymbol{\mu}_k^p &+ \mathbf{C}_{\mathbf{x}_k\mathbf{y}_k} \sum_{m=1}^{N} (\mathbf{C}_k^y)^{-1}(m)(\boldsymbol{\mu}_k^{ym} - \boldsymbol{\mu}_k^y(m)) \\
&+ \mathbf{C}_{\mathbf{x}_k\mathbf{y}_k} \sum_{m=1}^{N} (\mathbf{C}_k^y)^{-1}(m)\mathbf{C}_k^{ym}\mathbf{C}^{+}_{\mathbf{x}_k\mathbf{y}_k^m}(\boldsymbol{\mu}_k^{em} - \boldsymbol{\mu}_k^{pm}),
\end{aligned} \tag{35}$$

where

$$\begin{aligned}
\boldsymbol{\mu}_k^y &= (\boldsymbol{\mu}_k^y(1), \ldots, \boldsymbol{\mu}_k^y(N)), \\
\boldsymbol{\mu}_k^y(m) &= \mathbb{E}(\mathbf{y}_k^m|\mathbf{y}_{1:k-1}),
\end{aligned}$$

and

$$(\mathbf{C}_k^y)^{-1} = [(\mathbf{C}_k^y)^{-1}(1), (\mathbf{C}_k^y)^{-1}(2), \ldots, (\mathbf{C}_k^y)^{-1}(N)]$$

is an appropriate partition of matrix $(\mathbf{C}_k^y)^{-1}$ *such that Equation* (35) *holds. The superscript "+" stands for pseudoinverse.*

Proof. According to Equation (15), the mean of centralized fused state is given by

$$\boldsymbol{\mu}_k^e = \boldsymbol{\mu}_k^p + \mathbf{C}_{\mathbf{x}_k\mathbf{y}_k}(\mathbf{C}_k^y)^{-1}(\mathbf{y}_k - \boldsymbol{\mu}_k^y) \tag{36}$$

$$= \boldsymbol{\mu}_k^p - \mathbf{C}_{\mathbf{x}_k\mathbf{y}_k}(\mathbf{C}_k^y)^{-1}\boldsymbol{\mu}_k^y + \mathbf{C}_{\mathbf{x}_k\mathbf{y}_k}(\mathbf{C}_k^y)^{-1}\mathbf{y}_k \tag{37}$$

$$= \boldsymbol{\mu}_k^p - \mathbf{C}_{\mathbf{x}_k\mathbf{y}_k}(\mathbf{C}_k^y)^{-1}\boldsymbol{\mu}_k^y + \mathbf{C}_{\mathbf{x}_k\mathbf{y}_k} \sum_{m=1}^{N} (\mathbf{C}_k^y)^{-1}(m)\mathbf{y}_k^m. \tag{38}$$

Based on the assumption that all $C_{x_k y_k^m}$ have full column rank, we obtain

$$\mathbf{C}^{+}_{\mathbf{x}_k\mathbf{y}_k^m}\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k^m} = \mathbf{I}, \quad m = 1, \ldots, N. \tag{39}$$

Thus, combining Equations (37) and (38) and Equation (39) yields

$$\begin{aligned}&\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k}(\mathbf{C}_k^y)^{-1}\mathbf{y}_k\\=&\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k}\sum_{m=1}^{N}(\mathbf{C}_k^y)^{-1}(m)\mathbf{y}_k^m\\=&\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k}\sum_{m=1}^{N}(\mathbf{C}_k^y)^{-1}(m)\mathbf{C}_k^{ym}\mathbf{C}^{+}_{\mathbf{x}_k\mathbf{y}_k^m}\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k^m}(\mathbf{C}_k^{ym})^{-1}\mathbf{y}_k^m.\end{aligned}\tag{40}$$

In order to obtain the centralized estimation mean, that is, fuse the mean of local sensor state estimation at the fusion center, we use Equations (32) and (40) to eliminate $\mathbf{y}_k$ from Equation (36). From Equation (32), we have

$$\begin{aligned}&\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k^m}(\mathbf{C}_k^{ym})^{-1}\mathbf{y}_k^m\\=&\boldsymbol{\mu}_k^{em}-\boldsymbol{\mu}_k^{pm}+\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k^m}(\mathbf{C}_k^{ym})^{-1}\boldsymbol{\mu}_k^{ym}.\end{aligned}\tag{41}$$

Thus, substituting Equation (41) into Equation (40) and recalling Equations (36)–(38), we have

$$\begin{aligned}\boldsymbol{\mu}_k^e=&\boldsymbol{\mu}_k^p-\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k}(\mathbf{C}_k^y)^{-1}\boldsymbol{\mu}_k^y\\&+\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k}\sum_{m=1}^{N}(\mathbf{C}_k^y)^{-1}(m)\mathbf{C}_k^{ym}\mathbf{C}^{+}_{\mathbf{x}_k\mathbf{y}_k^m}\\&\times(\boldsymbol{\mu}_k^{em}-\boldsymbol{\mu}_k^{pm}+\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k^m}(\mathbf{C}_k^{ym})^{-1}\boldsymbol{\mu}_k^{ym})\\=&\boldsymbol{\mu}_k^p+\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k}\sum_{m=1}^{N}(\mathbf{C}_k^y)^{-1}(m)(\boldsymbol{\mu}_k^{ym}-\boldsymbol{\mu}_k^y(m))\\&+\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k}\sum_{m=1}^{N}(\mathbf{C}_k^y)^{-1}(m)\mathbf{C}_k^{ym}\mathbf{C}^{+}_{\mathbf{x}_k\mathbf{y}_k^m}(\boldsymbol{\mu}_k^{em}-\boldsymbol{\mu}_k^{pm}).\end{aligned}$$

Therefore, we obtain the state mean of the distributed estimation fusion. □

Remark 4. *The proposed distributed estimation fusion formula is equivalent to the centralized estimation fusion, therefore, the two fusion methods have the same fusion performance. However, distributed estimation fusion is more desirable in real applications in terms of reliability, survivability, communication bandwidth and computational burdens. From Equation (35), we can see that* $\boldsymbol{\mu}_k^p$, $\mathbf{C}_k^p$, $\boldsymbol{\mu}_k^y$, $\mathbf{C}_k^y$, $\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k}$ *can be calculated in advance and the terms* $\boldsymbol{\mu}_k^{pm}$, $\boldsymbol{\mu}_k^{ym}$, $\boldsymbol{\mu}_k^{em}$, $\mathbf{C}_k^{ym}$, $\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k^m}$ *need to be transmitted by local sensors in real time.*

Remark 5. *From the update step (30) of the centralized estimation fusion with GP-UKF, the distributed estimation fusion method (35) is also suitable for the GP-UKF. Since the detailed description and proof for the distributed GP-UKF fusion are similar to Proposition 1, we omit the results. Note that the distributed GP-UKF fusion is different from the distributed GP-ADF fusion in the prediction step.*

4. Numerical Examples

In this section, we assess the performance of our fusion methods with two different examples.

4.1. 1D Example

We consider the nonlinear dynamic system with two sensor measurements as follows:

$$\mathbf{x}_{k+1}=0.5\mathbf{x}_k+\frac{25\mathbf{x}_k}{1+\mathbf{x}_k^2}+\mathbf{w}_k,\quad \mathbf{w}_k\sim\mathcal{N}(0,\sigma^2)\tag{42}$$

$$\mathbf{y}_k^m=5\sin(2\mathbf{x}_k+\mathbf{z}^m)+\mathbf{v}_k^m,\quad \mathbf{v}_k^m\sim\mathcal{N}(0,0.1^2), m=1,2,\tag{43}$$

where $\mathbf{z}^m, m = 1, 2$ are given by $\mathbf{z}^1 = 0$ and $\mathbf{z}^2 = \frac{\pi}{4}$, respectively. This nonlinear system is popularly used in many papers (see References [7,10]) to measure the performance of nonlinear filters. We regard the first 200 samples as the training data and use the rest 50 samples called testing data to test the fusion methods. Assume that the $\mathbf{x}_{-200}$ is a Gaussian distribution with prior mean $\mu_{-200} = 0$ and variance $\sigma^2_{-200} = 1.5^2$. The estimation performance of the single sensor and multisensor fusion is evaluated by the average Mahalanobis distances that are also used in Reference [7]. The Mahalanobis distance, which is defined between the ground truth and the filtered mean, is as follows:

$$\text{Maha} = \sqrt{(\mathbf{x}_k - \hat{\mathbf{x}}_k)^{\mathrm{T}} (\mathbf{C}_k^e)^{-1} (\mathbf{x}_k - \hat{\mathbf{x}}_k)}, \tag{44}$$

where $\mathbf{C}_k^e$ is the estimated covariance. For the Mahalanobis distance, lower values indicate better performance [7]. Figures 2 and 3 show the average Mahalanobis distances of the single filter with GP-ADF and GP-UKF, and the fusion methods after 1000 independent runs. Figure 4 shows the average Mahalanobis distances for different system noises.

According to the average Mahalanobis distances in Figures 2 and 3, we can see that the estimation performance of Sensor 2 is better than that of Sensor 1. Furthermore, the multisensor estimation fusion methods perform better than the single sensor filters with GP-ADF and GP-UKF, respectively. Thus the effectiveness and advantages of the multisensor estimation fusion with Gaussian process can be observed. In addition, GP-ADF fusion performs better than GP-UKF fusion for the system noise $\sigma^2 = 1$ and GP-UKF fusion performs better than GP-ADF fusion for the system noise $\sigma^2 = 2$. It means that GP-ADF fusion and GP-UKF fusion have their comparative advantages for different system noises. It suggests that GP-ADF fusion may be a better choice for the small system noise and GP-UKF fusion may be worth considering for the relatively large system noise. From Figure 4, we can see that the proposed fusion methods enjoy better performance for different system noises. It demonstrates the consistence of our fusion methods for different system noises.

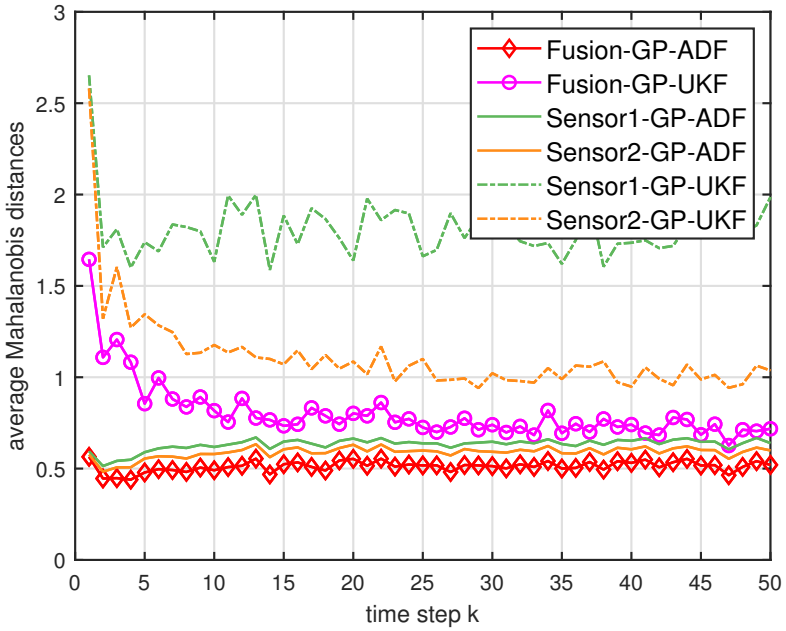

Figure 2. The average Mahalanobis distances of the single sensor and multisensor fusion with $\sigma^2 = 1$.

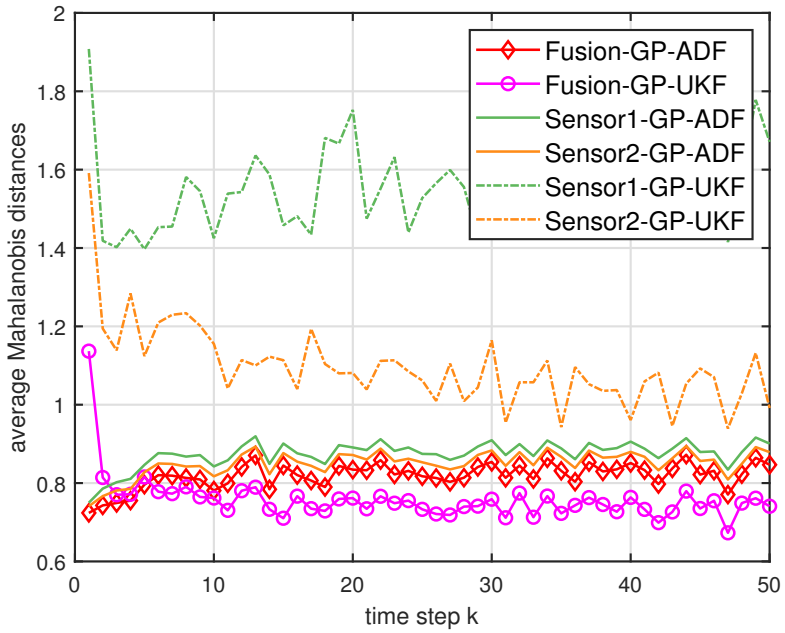

Figure 3. The average Mahalanobis distances of the single sensor and multisensor fusion with $\sigma^2 = 2$.

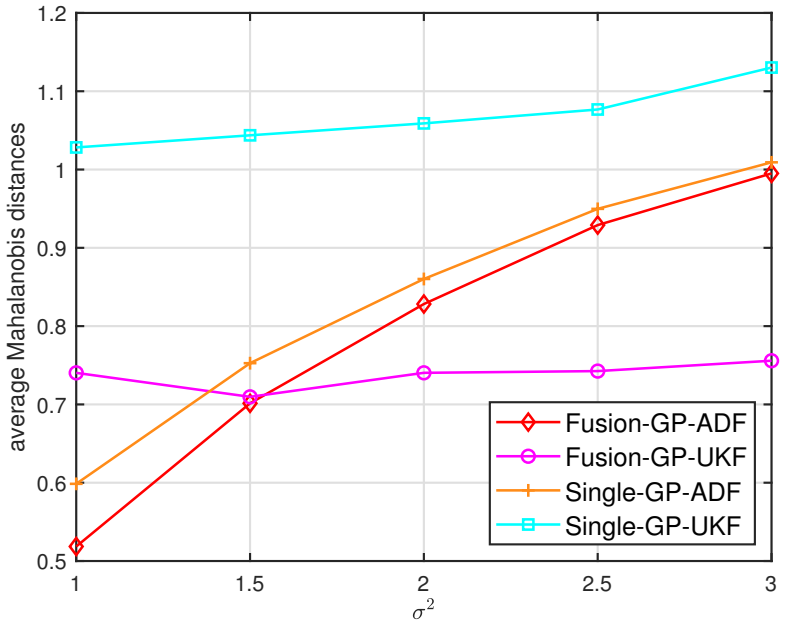

Figure 4. The average Mahalanobis distances for different system noises σ^2.

4.2. 2D Nonlinear Dynamic System

In this subsection, we elaborate the performance of the fusion methods with GP-ADF and GP-UKF, and we consider the example with constant turn motion model in target tracking. The root mean square error (RMSE) is used as the estimation performance measure. It is defined as follows:

$$\text{RMSE}_k = \sqrt{\frac{1}{M}\sum_{j=1}^{M} \|\mathbf{x}_k^j - \hat{\mathbf{x}}_k^j\|_2^2}, \tag{45}$$

where M is the total number of simulation runs, $\mathbf{x}_k^j$ is the true simulated state for the jth simulation, and $\hat{\mathbf{x}}_k^j$ is the estimated state value for the jth simulation at time k, $j = 1, \ldots, M$. The lower the value of the RMSE is, the better the performance of corresponding method is.

4.2.1. Experimental Setup

We consider the constant turn motion model with two sensors as follows:

$$\mathbf{x}_{k+1} = \mathbf{F}\mathbf{x}_k + \mathbf{w}_k, \quad \mathbf{w}_k \sim \mathcal{N}(0, \mathbf{Q}_k), \tag{46}$$

$$\mathbf{y}_k^m = \begin{bmatrix} \sqrt{(\mathbf{x}_k(1) - \mathbf{z}^m(1))^2 + (\mathbf{x}_k(2) - \mathbf{z}^m(2))^2} \\ arctan\frac{\mathbf{x}_k(2) - \mathbf{z}^m(2)}{\mathbf{x}_k(1) - \mathbf{z}^m(1)} \end{bmatrix} + \mathbf{v}_k^m,$$

$$\mathbf{v}_k^m \sim \mathcal{N}(0, \mathbf{R}_k), \ \ m = 1, 2, \tag{47}$$

where the state transition matrix $\mathbf{F}$ is given by

$$\mathbf{F} = \begin{bmatrix} \cos\Omega & \sin\Omega \\ -\sin\Omega & \cos\Omega \end{bmatrix}$$

with angular velocity Ω, the covariance matrix of transition noise satisfies

$$\mathbf{Q}_k = \begin{bmatrix} 5 & 0 \\ 0 & 5 \end{bmatrix},$$

and the covariance matrix of measurement noise is

$$\mathbf{R}_k = \begin{bmatrix} 5^2 & 0 \\ 0 & (\frac{0.1*\pi}{180})^2 \end{bmatrix}.$$

The sensor positions are given by $\mathbf{z}^1 = [-200, 200]^{\mathrm{T}}$ and $\mathbf{z}^2 = [200, -200]^{\mathrm{T}}$, respectively.

We use the transition function (46) and measurement functions (47) to simulate a group of data. The values of angular velocity Ω are given by $\Omega = \frac{2\pi}{90}$ rad/s and $\Omega = 0.5$ rad/s, respectively, which correspond to the small turn motion model and large turn motion model. The first κ samples are regarded as the training data and the rest 50 samples called testing data are used to test the fusion methods. The $\{\mathbf{x}_\tau, \mathbf{y}_\tau^1, \mathbf{y}_\tau^2\}_{\tau=-\kappa}^{-1}$ are the true simulated state and two-sensor observations used to train the models. Assume that the $\mathbf{x}_{-\kappa}$ is a Gaussian distribution with the prior mean $\boldsymbol{\mu}_{-\kappa} = [0, 0]^{\mathrm{T}}$ and the identity matrix variance $\mathbf{S}_{-\kappa} = \mathbf{I}$. The initial value of filtering for each sensor is set as the Cartesian coordinate transformation value based on the Polar coordinate observation with a random perturbation for each dimension. We use the random perturbation $N(10, 1)$ for the $\kappa = 300$ training data case and $N(20, 1)$ for the $\kappa = 30, 60$ training data cases, respectively. The initial value of fusion filtering is the mean of the initial values of all sensors. Based on the available observation information, the outputs of GP-ADF and GP-UKF for each single sensor are the mean and covariance terms $\boldsymbol{\mu}_k^{pm}$, $\mathbf{C}_k^{pm}$, $\boldsymbol{\mu}_k^{em}$, $\mathbf{C}_k^{em}$, $\boldsymbol{\mu}_k^{ym}$, $\mathbf{C}_k^{ym}$, $\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k^m}$. The distributed fusion methods, the RCC-CI algorithm [37] and the convex combination method [43], are also used as a comparison with our distributed fusion method. The RCC-CI algorithm and the convex combination method directly use the estimated mean and covariance matrix to fuse the estimated results. Our methods take full advantages of the results of the single sensor Gaussian process filters such as the cross term $\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k^m}$ which is easily obtained. Note that all of the fusion methods are based on the local estimates that are distilled from the measurements. Thus, the comparison is fair from the available observation information. The simulation results of the RCC-CI algorithm are based on the YALMIP toolbox [55]. We compare the fusion methods with the RCC-CI algorithm and the convex combination method by RMSE after $M = 1000$ independent simulation runs. Meanwhile, we also compare the computation time of the multisensor estimation fusion methods with the RCC-CI algorithm and the convex combination method. The ratio of full rank, defined as the percentage of full column rank of the cross term $\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k^m}$ for the single sensor filtering at every time step after $M = 1000$ independent runs, is used to test the condition of equivalence. If the

ratio of full rank is less than 100%, the condition is broken. The simulations are done under Matlab (MathWorks, Inc., Natick, MA, USA) R2018b with ThinkPad W540.

Figure 5 describes the ratios of full column rank in the single sensor case for testing data with $\kappa = 300$ training data and $\Omega = \frac{2\pi}{90}, 0.5$ rad/s. The RMSEs for $\Omega = \frac{2\pi}{90}, 0.5$ rad/s in the case of $\kappa = 300$ training data are depicted in Figures 6 and 7, respectively. The average computation time of these fusion methods for $\Omega = \frac{2\pi}{90}, 0.5$ rad/s in the case of $\kappa = 300$ training data after 1000 independent simulation runs are depicted in Figures 8 and 9, respectively. The diamond line represents the RMSE of the centralized estimation fusion with GP-ADF (CMF-GP-ADF) and the star line represents the distributed estimation fusion with GP-ADF (DMF-GP-ADF). The circle line represents the RMSE of the centralized estimation fusion with GP-UKF (CMF-GP-UKF) and the x line represents the distributed estimation fusion with GP-UKF (DMF-GP-UKF). The upper triangle line represents the RMSE of the RCC-CI fusion algorithm with GP-ADF (RCC-CI-GP-ADF) and the lower triangle line represents the RMSE of the RCC-CI fusion algorithm with GP-UKF (RCC-CI-GP-UKF). The square line represents the RMSE of the convex combination method, that is, covariance weight, with GP-ADF (CW-GP-ADF) and the + line represents the RMSE of the convex combination method with GP-UKF (CW-GP-UKF). At the same time, solid line is used for the single sensor GP-ADF filter and dash-dotted line is used for the single GP-UKF filter. Similarly, the ratios of full column rank in the single sensor case for testing data with $\kappa = 30, 60$ training data and $\Omega = \frac{2\pi}{90}$ rad/s are depicted in Figures 10 and 11, respectively. The RMSEs for $\Omega = \frac{2\pi}{90}$rad/s in the case of $\kappa = 30, 60$ training data are depicted in Figures 12 and 13, respectively. The average computation time of these fusion methods for testing data with $\kappa = 30, 60$ training data and $\Omega = \frac{2\pi}{90}$ rad/s after 1000 independent simulation runs are depicted in Figures 14 and 15, respectively.

4.2.2. Experimental Analysis

We divide the experimental analysis into two cases, that is, the equivalence condition is satisfied or not. Some phenomena and analysis are described as follows:

Case 1: the equivalence condition is satisfied.

- From Figure 5, we can see that the ratios of full rank are all equal to 100%. It means that the cross terms of the single sensor filters are all full column rank for the $\kappa = 300$ training data case and thus the equivalence condition of centralized fusion and distributed fusion is satisfied in Proposition 1. Meanwhile, from Figures 6 and 7, the RMSE of the distributed estimation fusion is the same as that of the centralized estimation fusion based on GP-ADF and GP-UKF, respectively. It demonstrates the equivalence between the centralized estimation fusion and the distributed estimation fusion in Proposition 1 under the condition of full column rank.
- From Figures 6 and 7, it can be seen that the RMSE of the multisensor estimation fusion method is lower than that of the single sensor filtering. It shows that the multisensor fusion improves the estimation accuracy.
- In addition, the RMSE of multisensor estimation fusion method is lower than that of the RCC-CI algorithm and the convex combination method. It implies the effectiveness of the fusion methods based on Gaussian processes. The possible reason is that our estimation fusion methods extract more extra correlation information, and the RCC-CI algorithm and the convex combination method only use the local estimates with mean and covariance.
- Compared Figure 6 with Figure 7, we can find that our methods do well in the different angular velocity cases, i.e., the small turn motion model and large turn motion model. It confirms that our fusion methods have fine applicability with better performance.
- From Figures 8 and 9, we can find that the computation time of the distributed estimation fusion is less than that of the centralized estimation fusion. It demonstrates the superiority of the distributed estimation fusion under the same fusion performance. The computation time of the proposed fusion methods is much less than that of the RCC-CI algorithm. The possible reason is

due to solve a optimization problem for the RCC-CI algorithm. The convex combination method takes the least computation time, since it directly uses the weight combination with covariance.

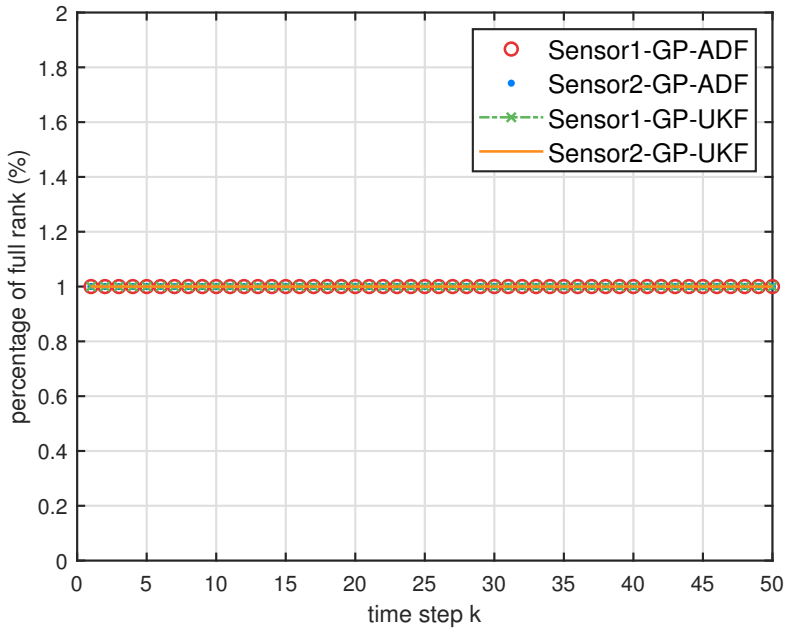

Figure 5. The ratios of full column rank of the cross term $\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k^m}$ in the single sensor case for testing data with $\kappa = 300$ training data and $\Omega = \frac{2\pi}{90}$ rad/s, 0.5 rad/s.

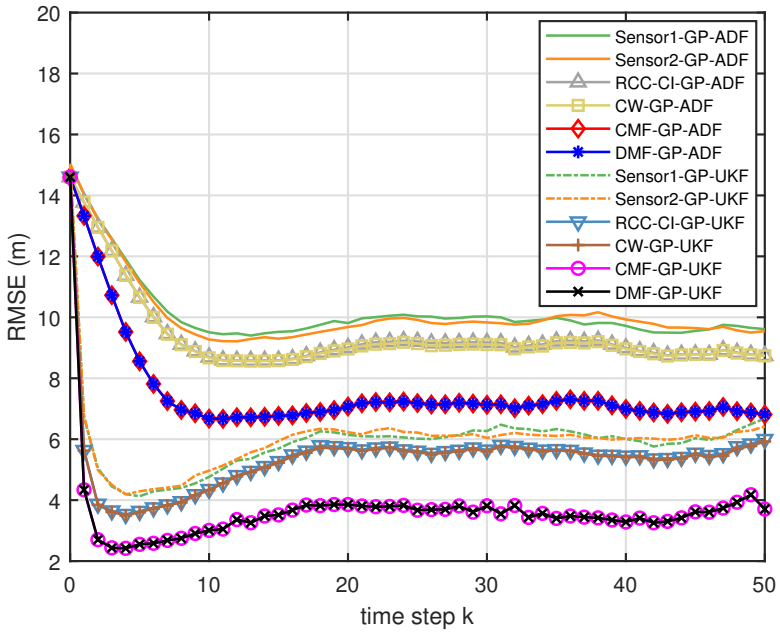

Figure 6. The RMSE for testing data with $\kappa = 300$ training data and $\Omega = \frac{2\pi}{90}$ rad/s.

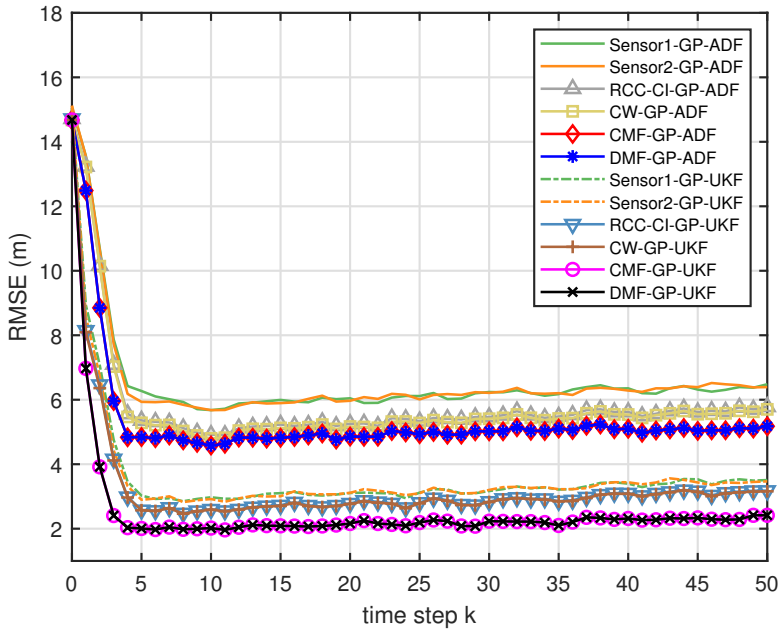

Figure 7. The RMSE for testing data with $\kappa = 300$ training data and $\Omega = 0.5$ rad/s.

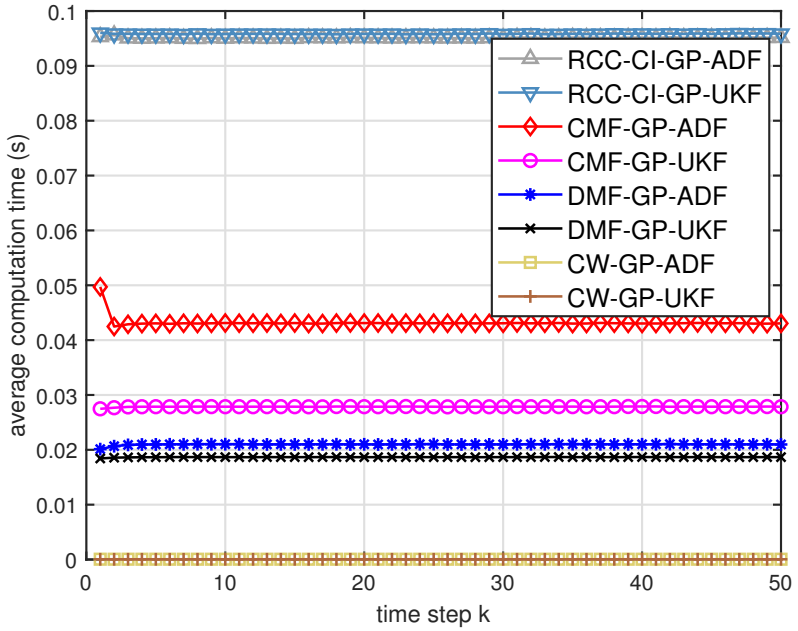

Figure 8. Average computation time of the three fusion methods for testing data with $\kappa = 300$ training data and $\Omega = \frac{2\pi}{90}$ rad/s.

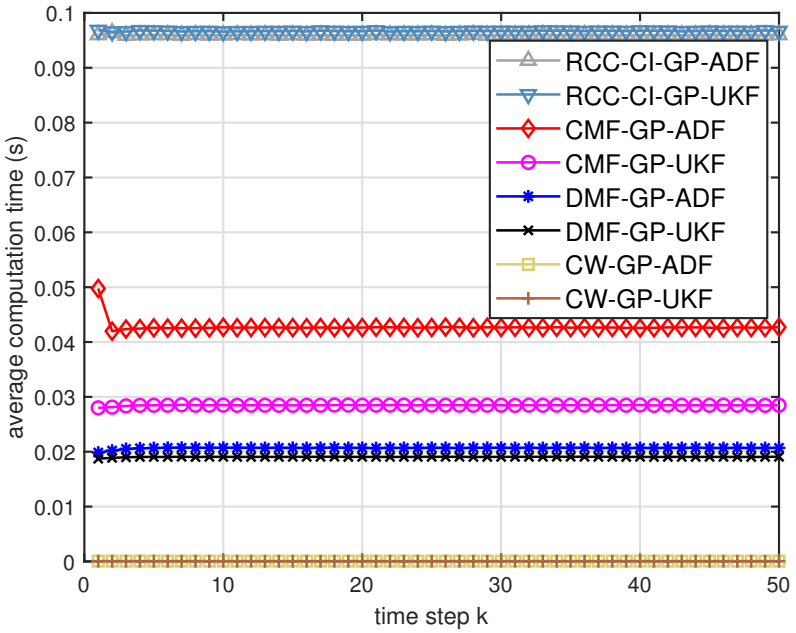

Figure 9. Average computation time of the three fusion methods for testing data with $\kappa = 300$ training data and $\Omega = 0.5$ rad/s.

Case 2: the equivalence condition is not satisfied.

- From Figures 10 and 11, it can be seen that the ratios of full rank are both less than 100% for the single sensor GP-UKF case with $\kappa = 30$ and $\kappa = 60$ training data. Thus, the equivalence between the centralized and distributed estimation fusion with GP-UKF is broken in Figures 12 and 13, respectively. The reason may be that the Gaussian models are relatively inaccurate with less training data, which can be known from the comparison with Figures 6, 12 and 13 for the same methods. At the same time, the finite-sample approximation of GP-UKF seriously depends on the Gaussian process models and the computation way of the cross terms is the sum about rank-one matrices for GP-UKF. However, the equivalence is still satisfied for the GP-ADF fusion. It implies GP-ADF is more stable than GP-UKF, which is also referred to in Reference [7].
- We can also see that the performance of GP-UKF fusion is better than that of GP-ADF fusion with $\kappa = 300$ training data from Figure 6 and a little worse with $\kappa = 30, 60$ training data from Figures 12 and 13. Meanwhile, from Figures 8, 14 and 15, the average computation time of GP-UKF fusion is less than that of GP-ADF fusion with $\kappa = 300$ training data and is contrary with $\kappa = 30, 60$ training data. It may inspire us that the GP-UKF fusion is suitable for the enough training data case and GP-ADF fusion does well in the small number of training data case for the turn motion systems.

In a word, distributed estimation fusion with GP-ADF is more stable, and has a better performance in the case of the small number of training data. If we have enough training data, GP-UKF fusion may be the better choice.

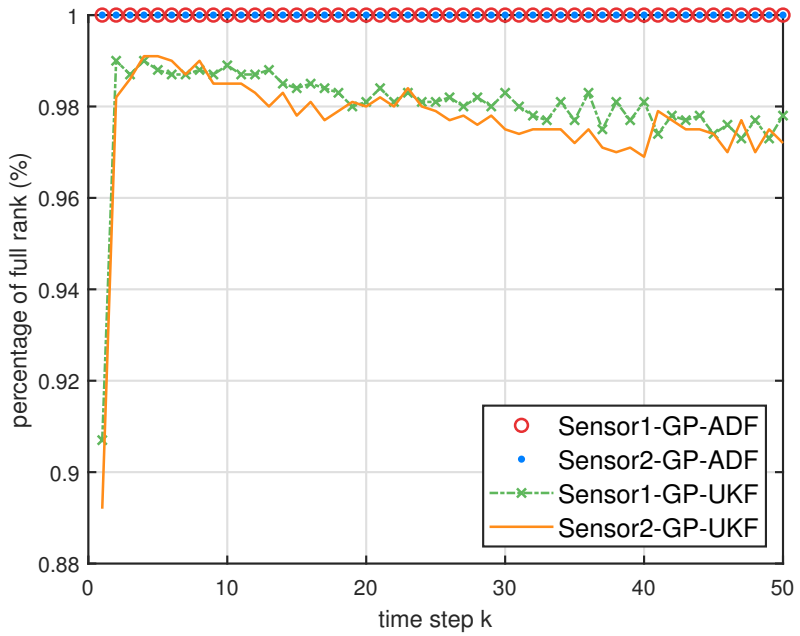

Figure 10. The ratios of full column rank of the cross term $\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k^m}$ in the single sensor case for testing data with $\kappa = 30$ training data and $\Omega = \frac{2\pi}{90}$ rad/s.

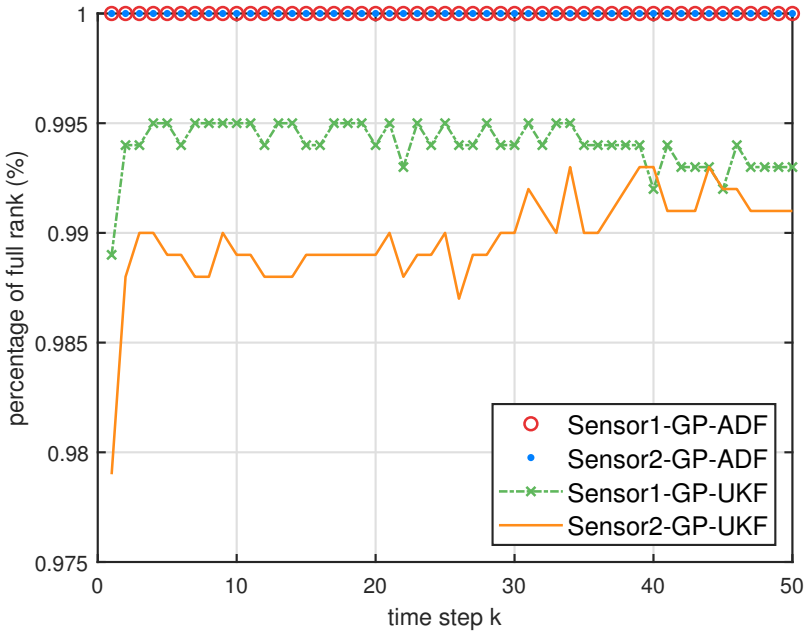

Figure 11. The ratios of full column rank of the cross term $\mathbf{C}_{\mathbf{x}_k\mathbf{y}_k^m}$ in the single sensor case for testing data with $\kappa = 60$ training data and $\Omega = \frac{2\pi}{90}$ rad/s.

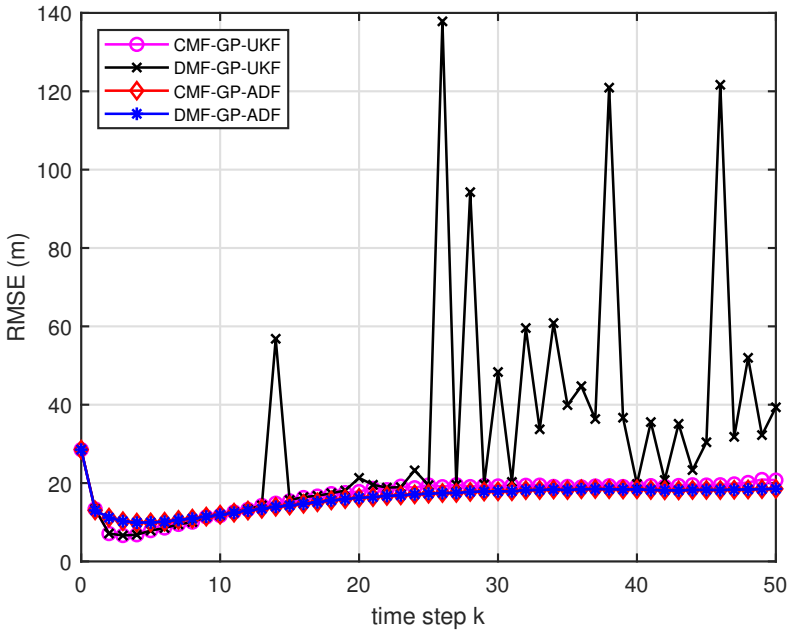

Figure 12. The RMSE for testing data with $\kappa = 30$ training data and $\Omega = \frac{2\pi}{90}$ rad/s.

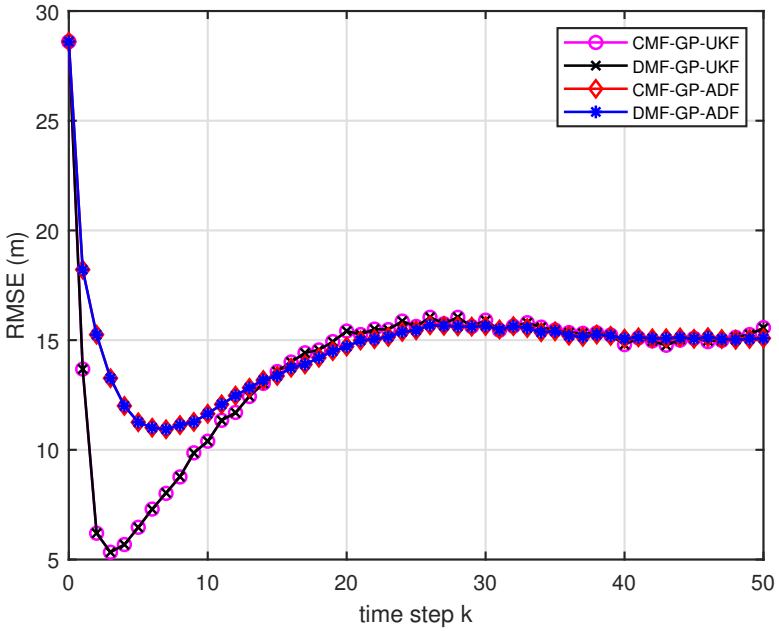

Figure 13. The RMSE for testing data with $\kappa = 60$ training data and $\Omega = \frac{2\pi}{90}$ rad/s.

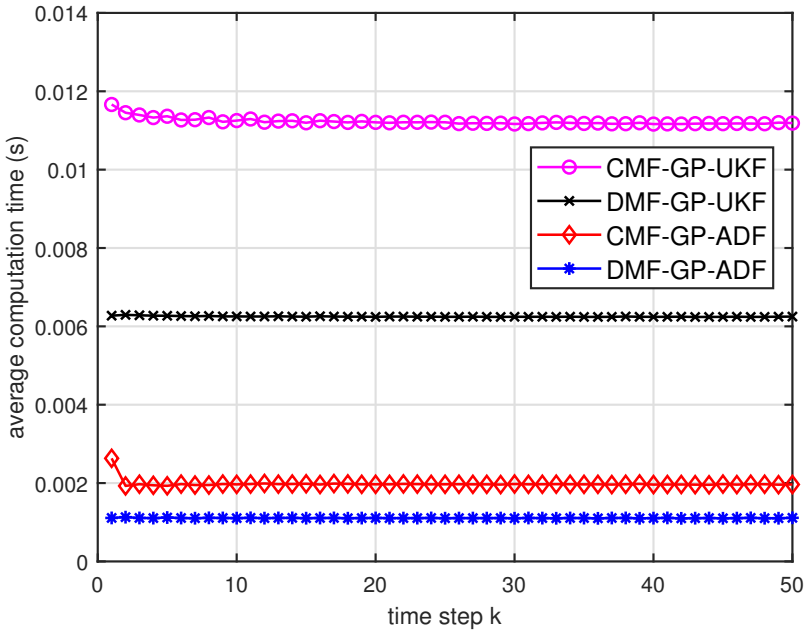

Figure 14. Average computation time of the three fusion methods for testing data with $\kappa = 30$ training data and $\Omega = \frac{2\pi}{90}$ rad/s.

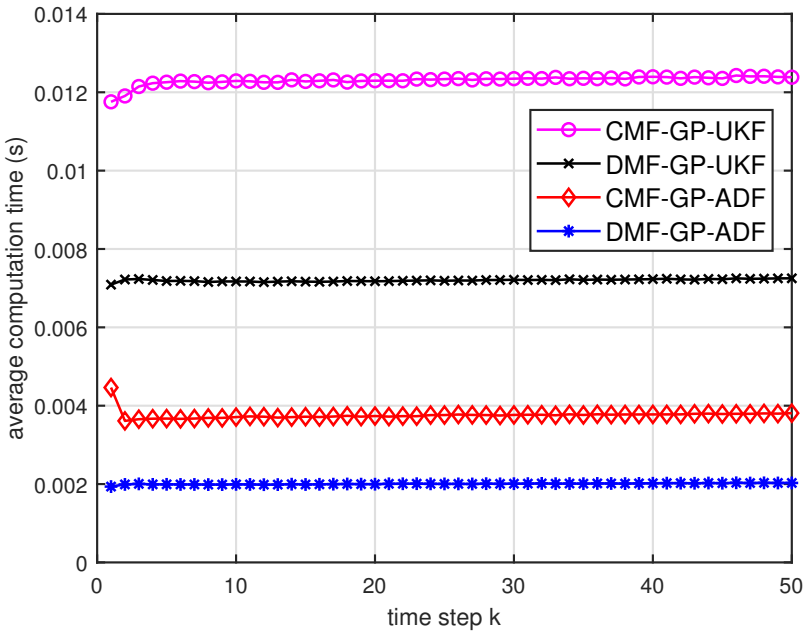

Figure 15. Average computation time of the three fusion methods for testing data with $\kappa = 60$ training data and $\Omega = \frac{2\pi}{90}$ rad/s.

5. Conclusions

In the context of this paper, the property matters if the real system does not closely follow idealized models or a parametric model cannot easily be determined for nonlinear systems. A non-parametric method, the Gaussian process, is introduced to learn models from training data, since it takes both model uncertainty and sensor measurement noise into account. In order to estimate the state of the multisensor nonlinear dynamic systems, we have used the Gaussian process models as the prior of the transition and measurement function of the dynamic system. Then the transition function and measurement function have been trained with Gaussian processes, respectively. Based on

the Gaussian process models for all available local sensor measurements, we have developed two fusion methods, centralized estimation fusion and distributed estimation fusion, with GP-ADF and GP-UKF, respectively. Taking full advantage of the nature of the Gaussian process, the equivalence between centralized estimation fusion and distributed estimation fusion has been derived under mild conditions. Simulations show that the equivalence is satisfied under given conditions and the multisensor estimation fusion performs better than the single sensor filters. Compared with the RCC-CI algorithm, the multisensor estimation fusion methods not only have higher accuracy, but also require less computation time. The estimation performance of multisensor estimation fusion methods is also better than that of the convex combination method. Future work may involve state initialization, Gaussian process latent variable models without ground truth states, multiple model fusion with different Gaussian processes for maneuvering target tracking and validate the methods with the real sensor data.

Author Contributions: Conceptualization, Y.L. and J.X.; Gaussian process methodology, Z.W. and Y.L.; fusion methodology, X.S. and Y.L.; data curation, J.X. and Y.L.; software, Y.L. and J.X.; validation, Y.L. and Z.W.; formal analysis, Y.L.; writing–original draft preparation, J.X. and Y.L.; writing–review and editing, Y.L.; supervision, X.S.; project administration, X.S.; funding acquisition, X.S.

Funding: This work was supported in part by the NSFC under Grant 61673282, the PCSIRT under Grant PCSIRT16R53 and the Fundamental Research Funds for the Central Universities under Grand No. 2012017yjsy140.

Acknowledgments: We would like to thank Yunmin Zhu for his insightful comments and helpful suggestions that greatly improved the quality of this paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Julier, S.J.; Uhlmann, J.K. Unscented filtering and nonlinear estimation. *Proc. IEEE* **2004**, *92*, 401–422. [CrossRef]
2. Lan, J.; Li, X.R. Multiple conversions of measurements for nonlinear estimation. *IEEE Trans. Signal Process.* **2017**, *65*, 4956–4970. [CrossRef]
3. Bar-Shalom, Y.; Li, X.R.; Kirubarajan, T. *Estimation with Applications to Tracking and Navigation: Theory, Algorithms and Software*; Wiley: New York, NY, USA, 2001.
4. Thrun, S.; Burgard, W.; Fox, D. *Probabilistic Robotics*; MIT Press: Cambridge, MA, USA, 2005.
5. Simon, D. *Optimal State Estimation: Kalman, H_∞, and Nonlinear Approaches*; Wiley-Interscience: New York, NY, USA, 2006.
6. Arulampalam, M.S.; Maskell, S.; Gordon, N.; Clapp, T. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Trans. Signal Process.* **2002**, *50*, 174–188. [CrossRef]
7. Deisenroth, M.P.; Huber, M.F.; Hanebeck, U.D. Analytic moment-based Gaussian process filtering. In Proceedings of the International Conference on Machine Learning, Montreal, QC, Canada, 14–18 June 2009.
8. Rasmussen, C.E.; Williams, C.K.I. *Gaussian Processes for Machine Learning*; MIT Press: Cambridge, MA, USA, 2006.
9. Huber, M.F. Nonlinear Gaussian Filtering: Theory, Algorithms, and Applications. Ph.D. Thesis, Karlsruhe Institute of Technology, Karlsruhe, Germany, 2015.
10. Deisenroth, M.P.; Turner, R.D.; Huber, M.F.; Hanebeck, U.D.; Rasmussen, C.E. Robust filtering and smoothing with Gaussian processes. *IEEE Trans. Automat. Control* **2012**, *57*, 1865–1871. [CrossRef]
11. Jacobs, M.A.; DeLaurentis, D. Distributed Kalman filter with a Gaussian process for machine learning. In Proceedings of the 2018 IEEE Aerospace Conference, Big Sky, MT, USA, 3–10 March 2018; pp. 1–12.
12. Guo, Y.; Li, Y.; Tharmarasa, R.; Kirubarajan, T.; Efe, M.; Sarikaya, B. GP-PDA filter for extended target tracking with measurement origin uncertainty. *IEEE Trans. Aerosp. Electron. Syst.* **2019**, *55*, 1725–1742. [CrossRef]
13. Preuss, R.; Von Toussaint, U. Global optimization employing Gaussian process-based Bayesian surrogates. *Entropy* **2018**, *20*, 201. [CrossRef]
14. Ko, J.; Fox, D. GP-BayesFilters: Bayesian filtering using Gaussian process prediction and observation models. *Autonomous Robots* **2009**, *27*, 75–90. [CrossRef]

15. Lawrence, N.D. Gaussian process latent variable models for visualisation of high dimensional data. In Proceedings of the NIPS'03 16th International Conference on Neural Information Processing Systems, Whistler, BC, Canada, 9–11 December 2003; pp. 329–336.
16. Wang, J.M.; Fleet, D.J.; Hertzmann, A. Gaussian process dynamical models. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 5–8 December 2005; pp. 1441–1448.
17. Ko, J.; Fox, D. Learning GP-BayesFilters via Gaussian process latent variable models. *Autonomous Robots* **2011**, *30*, 3–23. [CrossRef]
18. Csató, L.; Opper, M. Sparse on-line Gaussian processes. *Neural Comput.* **2002**, *14*, 641–668. [CrossRef]
19. Snelson, E.; Ghahramani, Z. Sparse Gaussian processes using pseudo-inputs. In Proceedings of the 18th International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 5–8 December 2005; pp. 1257–1264,
20. Seeger, M.; Williams, C.K.I.; Lawrence, N.D. Fast forward selection to speed up sparse Gaussian process regression. In Proceedings of the Workshop on AI and Statistics 9, Key West, FL, USA, 3–6 January 2003.
21. Smola, A.J.; Bartlett, P. Sparse greedy Gaussian process regression. In Proceedings of the Advances in Neural Information Processing Systems, Denver, CO, USA, 27 November–2 December 2000.
22. Quiñonero-Candela, J.; Rasmussen, C. A unifying view of sparse approximate Gaussian process regression. *J. Mach. Learn. Res.* **2005**, *6*, 1939–1959.
23. Velychko, D.; Knopp, B.; Endres, D. Making the coupled Gaussian process dynamical model modular and scalable with variational approximations. *Entropy* **2018**, *20*, 724. [CrossRef]
24. Yan, L.; Duan, X.; Liu, B.; Xu, J. Bayesian optimization based on K-optimality. *Entropy* **2018**, *20*, 594. [CrossRef]
25. Ko, J.; Klein, D.J.; Fox, D.; Hähnel, D. Gaussian processes and reinforcement learning for identification and control of an autonomous blimp. In Proceedings of the IEEE International Conference on Robotics and Automation, Roma, Italy, 10–14 April 2007.
26. Ko, J.; Klein, D.J.; Fox, D.; Haehnel, D. GP-UKF: Unscented Kalman filters with Gaussian process prediction and observation models. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, San Diego, CA, USA, 29 October–2 November 2007.
27. Ferris, B.; Hähnel, D.; Fox, D. Gaussian processes for signal strength-based location estimation. In Proceedings of the Robotics: Science and Systems, Philadelphia, PA, USA, 16–19 August 2006; pp. 1–8.
28. Maybeck, P.S. *Stochastic Models, Estimation, and Control*; Academic Press, Inc.: New York, NY, USA, 1979.
29. Boyen, X.; Koller, D. Tractable inference for complex stochastic processes. In Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence, Madison, WI, USA, 24–26 July 1998; pp. 33–42.
30. Opper, M. On-line learning in neural networks. In *ch. A Bayesian Approach to On-line Learning*; Cambridge University Press: Cambridge, UK, 1999; pp. 363–378.
31. Liggins, M.E.; Hall, D.L.; Llinas, J. *Handbook of Multisensor Data Fusion: Theory and Practice*; CRC Press: Boca Raton, FL, USA, 2009.
32. Bar-Shalom, Y.; Willett, P.K.; Tian, X. *Tracking and Data Fusion: A Handbook of Algorithms*; YBS Publishing: Storrs, CT, USA, 2011.
33. Shen, X.; Zhu, Y.; Song, E.; Luo, Y. Optimal centralized update with multiple local out-of-sequence measurements. *IEEE Trans. Signal Process.* **2009**, *57*, 1551–1562. [CrossRef]
34. Wu, D.; Zhou, J.; Hu, A. A new approximate algorithm for the Chebyshev center. *Automatica* **2013**, *49*, 2483–2488. [CrossRef]
35. Li, M.; Zhang, X. Information fusion in a multi-source incomplete information system based on information entropy. *Entropy* **2017**, *19*, 570. [CrossRef]
36. Gao, X.; Chen, J.; Tao, D.; Liu, W. Multi-sensor centralized fusion without measurement noise covariance by variational Bayesian approximation. *IEEE Trans. Aerosp. Electron. Syst.* **2011**, *47*, 718–272. [CrossRef]
37. Wang, Y.; Li, X.R. Distributed estimation fusion with unavailable cross-correlation. *IEEE Trans. Aerosp. Electron. Syst.* **2012**, *48*, 259–278. [CrossRef]
38. Chong, C.-Y.; Chang, K.; Mori, S. Distributed tracking in distributed sensor networks. In Proceedings of the American Control Conference, Seattle, WA, USA, 18–20 June 1986; pp. 1863–1868,
39. Zhu, Y.; You, Z.; Zhao, J.; Zhang, K.; Li, X.R. The optimality for the distributed Kalman filtering fusion with feedback. *Automatica* **2001**, *37*, 1489–1493. [CrossRef]

40. Li, X.R.; Zhu, Y.; Jie, W.; Han, C. Optimal linear estimation fusion–Part I: Unified fusion rules. *IEEE Trans. Inf. Theory* **2003**, *49*, 2192–2208. [CrossRef]
41. Duan, Z.; Li, X.R. Lossless linear transformation of sensor data for distributed estimation fusion. *IEEE Trans. Signal Process.* **2010**, *59*, 362–372. [CrossRef]
42. Shen, X.J.; Luo, Y.T.; Zhu, Y.M.; Song, E.B. Globally optimal distributed Kalman filtering fusion. *Sci. China Inf. Sci.* **2012**, *55*, 512–529. [CrossRef]
43. Chong, C.-Y.; Mori, S. Convex combination and covariance intersection algorithms in distributed fusion. In Proceedings of the 4th International Conference on Information Fusion, Montreal, QC, Canada, 7–10 August 2001.
44. Sun, S.; Deng, Z.-L. Multi-sensor optimal information fusion Kalman filter. *Automatica* **2004**, *40*, 1017–1023. [CrossRef]
45. Song, E.; Zhu, Y.; Zhou, J.; You, Z. Optimal Kalman filtering fusion with cross-correlated sensor noises. *Automatica* **2007**, *43*, 1450–1456. [CrossRef]
46. Hu, C.; Lin, H.; Li, Z.; He, B.; Liu, G. Kullback–Leibler divergence based distributed cubature Kalman filter and its application in cooperative space object tracking. *Entropy* **2018**, *20*, 116. [CrossRef]
47. Bakr, M.A.; Lee, S. Distributed multisensor data fusion under unknown correlation and data inconsistency. *Sensors* **2017**, *17*, 2472. [CrossRef]
48. Xie, J.; Shen, X.; Wang, Z.; Zhu, Y. Gaussian process fusion for multisensor nonlinear dynamic systems. In Proceedings of the 37th Chinese Control Conference (CCC), Wuhan, China, 25–27 July 2018; pp. 4124–4129.
49. Osborne, M. Bayesian Gaussian Processes for Sequential Prediction, Optimisation and Quadrature. Ph.D. Thesis, University of Oxford, Oxford, UK, 2010.
50. Nguyen-Tuong, D.; Seeger, M.; Peters, J. Local Gaussian process regression for real time online model learning. In Proceedings of the International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 8–10 December 2008.
51. Deisenroth, M. *Efficient Reinforcement Learning using Gaussian Processes*; KIT Scientific Publishing: Karlsruhe, Germany, 2010.
52. Ghahramani, Z.; Rasmussen, C.E. Bayesian Monte Carlo. *Adv. Neural Inf. Process. Syst.* **2003**, *15*, 489–496.
53. Candela, J.Q.; Girard, A.; Larsen, J.; Rasmussen, C.E. Propagation of uncertainty in Bayesian kernel models - application to multiple-step ahead forecasting. *IEEE Int. Conf. Acoust. Speech Signal Process.* **2003**, *2*, 701–704.
54. Zhu, Y. *Multisensor Decision and Estimation Fusion*; Kluwer Academic Publishers: Boston, MA, USA, 2003.
55. Öfberg, J.L. Yalmip: A toolbox for modeling and optimization in matlab. In Proceedings of the CACSD Conference, Taipei, Taiwan, China, 2–4 September 2004.