IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 15 June 2022, accepted 15 July 2022, date of publication 21 July 2022, date of current version 28 July 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3193101

RESEARCH ARTICLE

Adaptive Kernel Learning Kalman Filtering With Application to Model-Free Maneuvering Target Tracking

YUANKAI LI[1,2], (Senior Member, IEEE), JIAXIN LOU[1,3], XIAOSU TAN[1], YANKE XU[4], JINPENG ZHANG[4], AND ZHONGLIANG JING[2], (Senior Member, IEEE)

[1]School of Aeronautics and Astronautics, University of Electronic Science and Technology of China, Chengdu 611731, China
[2]School of Aeronautics and Astronautics, Shanghai Jiao Tong University, Shanghai 200240, China
[3]Hubei Aerospace Flight Vehicle Institute, Wuhan 430040, China
[4]Luoyang Optoelectro Technology Development Center, Luoyang 471009, China

Corresponding author: Yuankai Li (yuankai.li@uestc.edu.cn)

This work was supported in part by the Aviation Science Foundation of China under Grant 20190108001, and in part by the Natural Science Foundation of China under Grant 51822502 and Grant 42074038.

ABSTRACT Kernel method is a non-parametric linearization method for system modeling, which uses nonlinear projection from input data space to high-dimensional Hilbert feature space and employs kernel function for hiding the projection operator in a linear learner to replace inner product calculation in Hilbert space and to avoid the curse of dimensionality. Kernel method is data-driven and learnable to nonlinear model. For independency to a priori model, it provides a novel way of state estimation for nonlinear dynamic systems with model uncertainties. In this paper, an adaptive kernel learning Kalman filtering method is proposed and applied into the problem of maneuvering target tracking. Without use of a priori system model, the method performs Kalman filtering in a reproducing kernel Hilbert space (RKHS) by estimating conditional embedding operator (CEO) as system state transfer function from training data. For the stochastic uncertainty in system model, the maximum correntropy criterion (MCC) is introduced to obtain kernel parameter optimization and balance of estimation performance, while a sliding window is designed for online updating estimation of state transfer function to get adaptability to unknown system dynamics. Such the construction for kernel Kalman filtering (KKF) is helpful to extend application from time-series signal processing to state estimation of uncertain dynamical systems. Simulation scenarios include average sunspot prediction, hovering target tracking and hypersonic maneuvering target tracking, corresponding to verification in low-dynamic, periodic-dynamic and high-dynamic systems. Numerical results have illustrated that the proposed adaptive KKF can realize model-free tracking for target that has nonlinear dynamical motion, shown adaptability to non-cooperative target maneuver and better precision and convergence speed than typical model-based algorithms.

INDEX TERMS Kernel Kalman filtering, adaptive filtering, kernel learning, Bayesian online learning, RKHS, conditional embedding operator, maximum correntropy criterion, maneuvering target tracking.

I. INTRODUCTION

Target tracking involves in many research areas, such as pattern recognition, artificial intelligence, and autonomous control, with promising applications on unmanned systems [1]–[3], feature classification and decision [4], reconnaissance and surveillance [5], biomedical engineering [6], human-machine interaction [7], etc.

The maneuvering target tracking technique is essentially a class of nonlinear pattern recognition problems [8], where a priori of the target motion is usually required for building a tracking model and based on which predicting the

The associate editor coordinating the review of this manuscript and approving it for publication was Min Wang.

current motional or featured states and rules of the target. However, modeling quality greatly affects the tracking effectiveness, and model mismatch or unmodeled dynamics probably lead to tracking divergence or even target loss.

Independent with a priori system model, kernel method may carry out feature recognition, prediction and estimation for input signals based on theory of statistical learning [9]. The method uses nonlinear projection from input data space to a linearized high-dimensional Hilbert feature space, or say kernel space, and employs kernel trick to perform high-dimensional function approximation in inner product space and further the kernel parameter learning and optimization. Without requirement of a priori signal structure or system uncertainty, kernel method has data-driven characteristics. Applied into the target tracking problem, it can help to keep high tracking quality for the target with nonlinear uncertain motion and to promote environmental adaptability of the tracking algorithm.

Recently, kernel method has gained fast development in nonlinear time-series prediction and estimation and learning, generating many nonlinear filtering methods in reproducing kernel Hilbert space (RKHS) where Mercer condition is satisfied, such as kernel least mean squares (KLMS) [10], [11], kernel affine projection algorithm (KAPA) [12], [13], kernel recursive least squares (KRLS) [14] and extended KRLS [15], [16]. These algorithms are generally used for processing time-series signals with certainty, but unsatisfied for widely-existed uncertain dynamic systems or hard to be applied into maneuvering target tracking problem where stochastic uncertainties exist.

To improve applicability to dynamic systems, Ralaivola *et al.* [18] combined kernel method with a Kalman filter and produced kernel Kalman filtering (KKF) algorithm, in which the measurement and state vectors are projected into a Hilbert feature space to process. To use in state space of dynamic system, Song *et al.* [19], [20] embedded conditional distribution into the kernel projection to feature space, providing theoretic support for dynamical modeling in RKHS. In [21], [22], kernel Bayesian and Kalman criteria based on least squares are derived to form a unified class of non-parametric Bayesian rules. In [23], Zhu *et al.*, by introducing the conditional embedding operator (CEO) into KKF, presented a KKF-CEO algorithm that is capable of model learning. Using the embedded measurement in kernel space as hidden state, the algorithm can real-time construct state space model in RKHS and describe the system unknown dynamics accurately. Dang *et al.* [24] further used the maximum correntropy criterion (MCC) to replace the usual LMS [23] for kernel optimization, enhancing algorithm robustness to the system stochastic disturbance such as heavy-tailed noises. MCC can seek for the best solution under the worst case and make decision without loss of high-order statistical information under insufficient prior knowledge. For its advantage to robustness in complex environment, MCC is paid great attention recently. It is designed on the Kalman filter [25], [26], promoting the adaptability to non-Gaussian noises, and on the Cubature Kalman filter (CKF) [27], [28], increasing the robustness to nonlinear complex noises and stochastic disturbances. Yang *et al.* [29] studied the MCC based on generalized Gaussian distribution for improving target tracking precision under non-Gaussian noises.

Kernel Kalman filtering shows a trend of applying the kernel method into dynamic systems. The KKF algorithms above are established based on off-line training data for recursive estimation and global optimization. The inner parameter of the kernel projection is generally insensitive to system uncertainty, leading to inherent filtering error and difficulty of recognizing the accurate change rule of system states. To establish the adaptability of the KKF to system uncertainty, in this paper, a sliding window adaptive (SWA) kernel learning Kalman filtering algorithm is proposed. The algorithm is developed in the RKHS-based kernel Bayesian filtering framework. On one hand, we employed CEO for online learning the state transfer operator, eliminating the dependency to the system model. On the other hand, MCC is applied for local optimization of the kernel projection, balancing the filtering performances, and sliding window is introduced to rolling update the learning of the state transfer operator, guaranteeing the global adaptability to the system uncertainty. Because linear noise that is defined in high-dimensional RKHS can represent nonlinear noise with wide distribution in input space, and noise distribution in RKHS is updated by training data from real-time measurements rather than by the assumed typical statistics, the algorithm is naturally adaptable to non-Gaussian noise environment.

In the paper, three types of numerical scenarios that are average sunspot prediction, hovering target tracking and hypersonic maneuvering target tracking, regarding as low-, periodic- and high-dynamic system respectively, are used for simulation verification. The results have shown that the proposed adaptive kernel learning filter can realize model-free tracking of nonlinear motional target with adaptability to the non-cooperative maneuver, achieving better precision and convergence speed than typical model-based methods.

The rest of this paper is organized as follows. In Section 2, the kernel projection with Mercer condition is presented, followed by CEO-based KKF framework in Section 3. The proposed adaptive kernel learning filter with designed MCC and sliding window is given in Section 4. Simulations are in Section 5 to verify the algorithm and Section 6 concludes the paper.

II. KERNEL PROJECTION TO RKHS

Kernel method is a non-parametric linearized modeling method, where system signals in original data space are nonlinearly projected into a high-dimensional RKHS and processed in this feature space. The prerequisite of the kernel method is to construct the kernel projection.

Define a nonlinear projection φ to build a kernel function κ. For any arbitrary $x, y \in \mathcal{R}$, they satisfy

$$\kappa(x, y) = \langle \varphi(x), \varphi(y) \rangle \tag{1}$$

where φ is the feature projection operator from input vector space $\mathcal{R}$ to feature vector space $\mathcal{F}$. When κ satisfies Mercer condition [30], [31], i.e., κ is continuous, symmetric and positive definite, it has

$$1)\ \forall\, x \in \mathcal{R}, \quad \kappa\,(x, y) \in \mathcal{F};$$

2) $\kappa\,(x, y)$ is reproducible, which means for function in $\mathcal{F}$ as

$$h\,(\cdot) = \sum\nolimits_i a_i \kappa\,(x_i, \cdot) \tag{2}$$

where a_i is the weight at the ith dimension, there exist

$$\langle h, \kappa\,(y, \cdot)\rangle = \sum\nolimits_i a_i \kappa\,(x_i, y) = h(y) \tag{3}$$

Then, κ is a reproducing kernel of $\mathcal{F}$. If $\mathcal{F}$ is a complete inner product space with reproducing kernel, $\mathcal{F}$ is known as the reproducing kernel Hilbert space.

In an infinity-dimensional RKHS, the reproducing kernel satisfying Mercer condition can be written as

$$\kappa\,(x, y) = \sum\nolimits_{i=1}^{\infty} \zeta_i \varphi_i(x) \varphi_i(y) \tag{4}$$

where ζ_i are nonnegative eigenvalues, then the projection operator φ can be constructed by

$$\varphi(x) = [\sqrt{\zeta_1}\varphi(x_1), \sqrt{\zeta_2}\varphi(x_2), \ldots] \tag{5}$$

Generally, φ is difficult to get an explicit form and we often use kernel function to replace inner product between projected functions in RKHS to fulfill nonlinear projection calculation, as

$$\varphi(x)^T \varphi(y) = \kappa\,(x, y) \tag{6}$$

so that the nonlinear function can be linearized into a high-dimensional space.

III. KERNEL KALMAN FILTERING IN DYNAMIC SYSTEM

For a dynamic system, kernel projection needs to project dynamic model or variable functions into RKHS for linearized processing. Naturally, the Kalman filter structure can be employed, forming kernel Kalman filtering method [18]. In KKF, state transfer operator, by embedding conditional distribution into kernel projection, is constructed and learned recursively with kernel function and training data in input space. Clearly, it is independent with any prior knowledge. Using conditional embedding operator, Kalman filtering can be established in RKHS, realizing model-free state estimation of dynamic systems.

A. CONDITIONAL EMBEDDING OPERATOR

For a dynamic system in noise environment, system model expresses the transition rule of stochastic state variables. Therefore, the probability distribution projected in RKHS is required to be represented so that the system state transition operator can be built in kernel space.

For stochastic variables $X, Y \in \mathcal{R}$, embedding their corresponding functions $f(X), g(Y) \in \mathcal{R}$ into a kernel projection can construct the following general definitions.

Definition 1: Expectation of Embedded Distribution

Denote the probability distribution of X in an input space $\mathcal{R}$ as P_X and that of X projected by kernel function $k_{\mathcal{F}}$ to an RKHS $\mathcal{F}$ as $\varphi(X)$. Then, the distribution function $f(X)$ embedded in $\mathcal{F}$ has expectation formed as

$$\begin{aligned} E_{\mathcal{F}}\left[f(X)\right] &= \int \langle f, \varphi(x)\rangle_{\mathcal{F}} dP_X(x) \\ &= \langle f, \int \varphi(x) dP_X(x)\rangle_{\mathcal{F}} \\ &= \langle f, E_{\mathcal{F}}\left[\varphi(X)\right]\rangle_{\mathcal{F}} \\ &= \langle f, \mu_X\rangle_{\mathcal{F}} \end{aligned} \tag{7}$$

where $\mu_X := E_{\mathcal{F}}[\varphi(X)]$ is defined as the embedded expectation of X in $\mathcal{F}$. In the RKHS $\mathcal{F}$, for $\varphi(X)$ satisfying $E\left[k_{\mathcal{F}}(X, X)\right] < \infty$, μ_X can be approximated using finite samples by

$$\hat{\mu_X} = \frac{1}{m} \sum\nolimits_{i=1}^{m} \varphi\,(x_i) \tag{8}$$

where $\mathcal{D}_X = \{x_1, \ldots, x_m\}$ is a training set with m samples. Actually, μ_X describes the relation between the expectations of X in input space and in feature space.

Definition 2: Covariance of Embedded Distribution

Consider that stochastic variables X and Y have probability distribution P_X and P_Y, respectively, in input space $\mathcal{R}$ with their joint probability distribution P_{XY}, and denote their projection operators by kernel function $k_{\mathcal{F}}$ to RKHS $\mathcal{F}$ as φ and ϕ. According to Definition 1, the embedded expectation of their joint function $f(X)g(Y)$ in $\mathcal{F}$ has the form as

$$\begin{aligned} E_{\mathcal{F}}\left[f(X)g(Y)\right] &= E\left[\langle f, \varphi(X)\rangle_{\mathcal{F}} \langle g, \phi(Y)\rangle_{\mathcal{F}}\right] \\ &= E\left[\langle f \otimes g, \varphi(X) \otimes \phi(Y)\rangle_{\mathcal{F}\otimes\mathcal{F}}\right] \\ &= \langle f \otimes g, E_{\mathcal{F}}\left[\varphi(X)\right] \otimes \phi(Y)\rangle_{\mathcal{F}\otimes\mathcal{F}} \\ &= \langle f \otimes g, C_{XY}\rangle_{\mathcal{F}\otimes\mathcal{F}} \\ &= \langle f, C_{XY} g\rangle_{\mathcal{F}} \end{aligned} \tag{9}$$

where $\otimes$ represents tensor product, and define

$$\begin{aligned} C_{XY} &= E\left[\varphi(X) \otimes \phi(Y)\right] \\ &= \int \varphi(X) \otimes \phi(Y) dP_{XY}(x, y) \end{aligned} \tag{10}$$

as the embedded cross-covariance of X and Y in $\mathcal{F}$. For given $k_{\mathcal{F}}$, C_{XY} is determined by $P_{XY}(x, y)$. Taking $X = Y$ yields embedded covariance of X as

$$\begin{aligned} C_{XX} &= E\left[\varphi(X) \otimes \varphi(X)\right] \\ &= \int \varphi(X) \otimes \varphi(X) dP_X(x) \end{aligned} \tag{11}$$

Similarly, C_{XY} can be approximated by using finite sample set $\mathcal{D}_{XY} = \{(x_1, y_1), \cdots, (x_m, y_m)\}$ and projected sample sets $\Upsilon = [\varphi\,(x_1), \cdots, \varphi\,(x_m)]$ and $\Phi = [\phi\,(y_1), \cdots, \phi\,(y_m)]$, as

$$\hat{\mathcal{C}_{XY}} = \frac{1}{m} \sum\nolimits_{i=1}^{m} \varphi\,(x_i) \otimes \phi\,(y_i) = \frac{1}{m} \Upsilon \Phi^T \tag{12}$$

Definition 3: Embedded Conditional Expectation

For arbitrary stochastic variables $X, Y \in \mathcal{R}$, denote their conditional probability distribution as $P_{Y|X}$ and project it into

RKHS $\mathcal{F}$ with kernel function $k_{\mathcal{F}}$. Define expectation of the distribution function $g(Y)$ under condition of X embedded in $\mathcal{F}$ as$E_{\mathcal{F}}[g(Y)|X]$, then we have [19]

$$\mathcal{C}_{XX}E_{\mathcal{F}}[g(Y)|X] = \mathcal{C}_{XY}g \tag{13}$$

According to Definition 1, for $X = x$, there is

$$\begin{aligned} E_{\mathcal{F}}\left[g(Y)|x\right] &= \int \langle g(y), \varphi(x)\rangle_{\mathcal{F}} dP_{Y|X}\ (y|x) \\ &= \langle E_{\mathcal{F}}\left[g(Y)|X\right], \varphi(x)\rangle_{\mathcal{F}} \\ &= \langle \mathcal{C}_{XX}^{-1}\mathcal{C}_{XY}g, \varphi(x)\rangle_{\mathcal{F}} \\ &= \langle g, \mathcal{C}_{YX}\mathcal{C}_{XX}^{-1}\varphi(x)\rangle_{\mathcal{F}} \\ &= \langle g, \mu_{Y|x}\rangle_{\mathcal{F}} \end{aligned} \tag{14}$$

where $\mu_{Y|x}$ is the conditional expectation of Y under X embedded in $\mathcal{F}$, expressed as

$$\mu_{Y|x} := \mathcal{C}_{YX}\mathcal{C}_{XX}^{-1}\varphi(x) \tag{15}$$

According to Definition 2 and (12), one can approximate $\mu_{Y|x}$ with finite samples. Taking its operator $\mathcal{U}_{Y|X}$ as the conditional embedding operator, abbreviated to CEO, we may calculate it by

$$\begin{aligned} \hat{\mathcal{U}}_{\Upsilon|X} &= \hat{\mathcal{C}_{YX}}\hat{\mathcal{C}_{XX}^{-1}} \\ &= \frac{1}{m}\Phi\mathrm{Y}^T\left(\frac{1}{m}\Upsilon\Upsilon^T + \varsigma\boldsymbol{I}\right)^{-1} \\ &= \Phi\mathrm{Y}^T\left(\Upsilon\Upsilon^T + \varsigma m\boldsymbol{I}_m\right)^{-1} \end{aligned} \tag{16}$$

where ς is the regular factor.

It is noticeable that for a kernel function in finite domain, i.e., $E\left[k_{\mathcal{F}}\right] < \infty$, $E_{\mathcal{F}}[g(Y)|X]\in\mathcal{F}$ is always formed, but not for in continuous domain [18], [32]. In that case, (16) can be replaced, by matrix inversion theory [33], with

$$\hat{\mathcal{U}}_{Y|X} = \Phi(\boldsymbol{K} + \varsigma m\boldsymbol{I}_m)^{-1}\Upsilon^T \tag{17}$$

where $\boldsymbol{K}$ represents $\Upsilon^T\Upsilon$ and $\boldsymbol{I}_m$ is an $m-$ dimensional unit matrix. Clearly, $\boldsymbol{K}$ can be obtained with kernel function for CEO calculation.

B. CEO-BASED KKF FOR STATE ESTIMATION

The CEO expresses the transition function of system states in RKHS. It can be used to realize optimal state estimation based on Kalman filtering in kernelized high-dimensional linear space, forming CEO-based kernel Kalman filter. In the filter, firstly we use training samples to learn the CEO, building system state model in RKHS, and then complete state prediction and recursive update in the kernel space.

1) MODEL LEARNING WITH CEO

To simplify description, consider a dynamic system that has linear measurement, shown as in Fig. 1, in which $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ are system state and measurement, and ξ_i and v_i are process and measurement noise, respectively. The nonlinear function

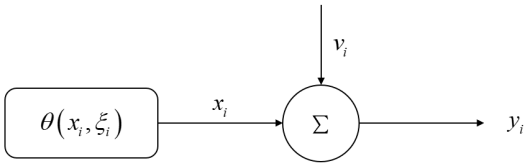

FIGURE 1. The dynamic system model in state space.

$\boldsymbol{\theta}$ represents the state transition operator, and the system can be expressed as

$$\boldsymbol{x}_{i+1} = \boldsymbol{\theta}\left(\boldsymbol{x}_i, \xi_i\right) \tag{18}$$

$$\boldsymbol{y}_i = \boldsymbol{x}_i + v_i \tag{19}$$

In the system having projected into RKHS, $\boldsymbol{x}_i$ becomes hidden states that are hard to obtain explicit expression or to be estimated with the usual Kalman filter. Hence, we need to use embedded measurement to construct the kernel projection of $\boldsymbol{\theta}$ as state transfer operator, and to realize recursive learning during measurement update.

Taking $\mathcal{D} = \left\{\boldsymbol{y}_1^0, \ldots, \boldsymbol{y}_{m+1}^0\right\}$ in $\boldsymbol{y}_i$ as the initial training data, according to (17), we may construct the state transfer operator in RKHS using CEO as

$$\boldsymbol{F}_i = \Phi(\boldsymbol{K} + \varsigma m\boldsymbol{I}_m)^{-1}\Upsilon^T \tag{20}$$

where $\Upsilon = \left[\varphi\left(\boldsymbol{y}_1^0\right), \ldots, \varphi\left(\boldsymbol{y}_m^0\right)\right]$, $\Phi = \left[\varphi\left(\boldsymbol{y}_2^0\right), \ldots, \varphi\left(\boldsymbol{y}_{m+1}^0\right)\right]$, and superscript 0 represents initial samples. $\boldsymbol{F}_i$ can connect the embedded state estimate, noted as $\hat{\mu_i}$, with one-step prediction of the embedded state, noted as $\hat{\mu_{i+1}}$, by

$$\hat{\mu_{i+1}} = \boldsymbol{F}_i\hat{\mu_i} \tag{21}$$

Then, the dynamic system can be modeled in RKHS with linearized state space form as

$$\hat{\mu_{i+1}} = \boldsymbol{F}_i\hat{\mu_i} + \hat{\xi}_i \tag{22}$$

$$\varphi\left(\boldsymbol{y}_i\right) = \boldsymbol{H}_i\hat{\mu_i} + \hat{v_i} \tag{23}$$

where $\hat{\xi}_i$ and $\hat{v_i}$ are corresponding noises in RKHS. $\boldsymbol{H}_i$ describes the measurement matrix, which can be transformed into unit matrix $\boldsymbol{I}$.

2) ONE-STEP ESTIMATION IN RKHS

In (22) and (23), $\hat{\xi}_i$ and $\hat{v_i}$ are infinite dimensional with limited power. Their statistic features can be determined by Definition 2 [13], [15], [18]. Considering that they have linearized form, we assume they are zero-mean Gaussian noises with covariance q and r, respectively, and take

$$\boldsymbol{Q}_i = q\boldsymbol{I} \tag{24}$$

$$\boldsymbol{R}_i = r\boldsymbol{I} \tag{25}$$

and $\boldsymbol{H}_i = \boldsymbol{I}$ for simplicity.

Given the initial measurement covariance and initial state covariance in RKHS with $\hat{\mu_0} = \varphi\left(\boldsymbol{y}_0\right)$ and $\boldsymbol{P}_0 = \varepsilon\boldsymbol{I}$, one-step estimation of a Kalman filter in kernel space has the following steps.

*Step*1: Calculate one-step prediction by

$$\hat{\mu_{i-}} = \boldsymbol{F}_{i-1}\hat{\mu_{i-1}} = \Phi(\boldsymbol{K} + \varsigma m\boldsymbol{I}_m)^{-1}\Upsilon^T\hat{\mu_{i-1}} = \Phi\boldsymbol{a}_i \tag{26}$$

where $\boldsymbol{a}_i = (\boldsymbol{K} + \varsigma m\boldsymbol{I}_m)^{-1}\Upsilon^T\hat{\mu_{i-1}}$.

*Step*2: Calculate one-step prediction covariance by

$$\begin{aligned}\boldsymbol{P_{i-}} &= \boldsymbol{F}_{i-1}\boldsymbol{P}_{i-1}\boldsymbol{F}_{i-1}^T + q\boldsymbol{I}\\ &= \Phi(\boldsymbol{K}+\varsigma m\boldsymbol{I}_m)^{-1}\Upsilon^T\boldsymbol{P}_{i-1}\Upsilon\left[(\boldsymbol{K}+\varsigma m\boldsymbol{I}_m)^{-1}\right]^T\Phi^T + q\boldsymbol{I}\\ &= \Phi\tilde{\boldsymbol{P}}_{i-}\Phi^T + q\boldsymbol{I}\end{aligned} \tag{27}$$

where $\tilde{\boldsymbol{P}}_{i-} = (\boldsymbol{K} + \varsigma m\boldsymbol{I}_m)^{-1}\Upsilon^T\boldsymbol{P}_{i-1}\Upsilon\left[(\boldsymbol{K} + \varsigma m\boldsymbol{I}_m)^{-1}\right]^T$.

*Step*3: Compute kernel Kalman filtering gain by

$$\begin{aligned}\boldsymbol{G}_i &= \boldsymbol{P_{i-}}\boldsymbol{H}_i^T\left[\boldsymbol{H}_i\boldsymbol{P_{i-}}\boldsymbol{H}_i^T + \boldsymbol{R}_i\right]^{-1}\\ &= \left(\Phi\tilde{\boldsymbol{P}}_{i-}\Phi^T + q\boldsymbol{I}\right)\left[\Phi\tilde{\boldsymbol{P}}_{i-}\Phi^T + (q+r)\boldsymbol{I}\right]^{-1}\\ &= \left[\frac{q}{q+r}\left[\Phi\tilde{\boldsymbol{P}}_{i-}\Phi^T + (q+r)\boldsymbol{I}\right] + \frac{r}{q+r}\Phi\tilde{\boldsymbol{P}}_{i-}\Phi^T\right]\\ &\quad\times\left[\Phi\tilde{\boldsymbol{P}}_{i-}\Phi^T + (q+r)\boldsymbol{I}\right]^{-1}\\ &= \frac{r}{q+r}\Phi\tilde{\boldsymbol{P}}_{i-}\Phi^T\left[\Phi\tilde{\boldsymbol{P}}_{i-}\Phi^T + (q+r)\boldsymbol{I}\right]^{-1} + \frac{q}{q+r}\boldsymbol{I}\end{aligned} \tag{28}$$

which can be reshaped, according to matrix theory, into iterative form as

$$\begin{aligned}\boldsymbol{G}_i &= \frac{r}{q+r}\Phi\left[(q+r)\boldsymbol{I}_m + \tilde{\boldsymbol{P}}_{i-}\Phi^T\Phi\right]^{-1}\tilde{\boldsymbol{P}}_{i-}\Phi^T + \frac{q}{q+r}\boldsymbol{I}\\ &= \frac{r}{q+r}\Phi\tilde{\boldsymbol{G}}_i\Phi^T + \frac{q}{q+r}\boldsymbol{I}\end{aligned} \tag{29}$$

where $\tilde{\boldsymbol{G}}_i = \left[(q+r)\boldsymbol{I}_m + \tilde{\boldsymbol{P}}_{i-}\Phi^T\Phi\right]^{-1}\tilde{\boldsymbol{P}}_{i-}$.

*Step*4: Update state estimation covariance by

$$\begin{aligned}\boldsymbol{P}_i &= (\boldsymbol{I} - \boldsymbol{H}_i\boldsymbol{G}_i)\boldsymbol{P_{i-}}\\ &= \Phi\left[\frac{r}{q+r}\tilde{\boldsymbol{P}}_{i-} - \frac{r}{q+r}\tilde{\boldsymbol{G}}_i\Phi^T\Phi\tilde{\boldsymbol{P}}_{i-} - \frac{qr}{q+r}\tilde{\boldsymbol{G}}_i\right]\\ &\quad\times\Phi^T + \left(q - \frac{q^2}{q+r}\right)\boldsymbol{I}\\ &= \Phi\tilde{\boldsymbol{P}}_i\Phi^T + \frac{qr}{q+r}\boldsymbol{I}\end{aligned} \tag{30}$$

where $\tilde{\boldsymbol{P}}_i = \frac{r}{q+r}\tilde{\boldsymbol{P}}_{i-} - \frac{r}{q+r}\tilde{\boldsymbol{G}}_i\Phi^T\Phi\tilde{\boldsymbol{P}}_{i-} - \frac{qr}{q+r}\tilde{\boldsymbol{G}}_i$.

*Step*5: Update state estimation by

$$\begin{aligned}\hat{\mu_i} &= \hat{\mu_{i-}} + \boldsymbol{G}_i\left(\varphi\left(\boldsymbol{y}_i\right) - \hat{\mu_{i-}}\right)\\ &= \Phi\boldsymbol{a}_i + \left(\frac{r}{q+r}\Phi\tilde{\boldsymbol{G}}_i\Phi^T + \frac{q}{q+r}\boldsymbol{I}\right)\left(\varphi\left(\boldsymbol{y}_i\right) - \Phi\boldsymbol{a}_i\right)\\ &= \Phi\left[\left(\frac{r}{q+r}\boldsymbol{I} - \frac{r}{q+r}\tilde{\boldsymbol{G}}_i\Phi^T\Phi\right)\boldsymbol{a}_i + \frac{r}{q+r}\tilde{\boldsymbol{G}}_i\Phi^T\varphi\left(\boldsymbol{y}_i\right)\right]\\ &\quad + \frac{q}{q+r}\varphi\left(\boldsymbol{y}_i\right)\\ &= \Phi\boldsymbol{b}_i + \frac{q}{q+r}\varphi\left(\boldsymbol{y}_i\right)\end{aligned} \tag{31}$$

where $\boldsymbol{b}_i = \left(\frac{r}{q+r}\boldsymbol{I} - \frac{r}{q+r}\tilde{\boldsymbol{G}}_i\Phi^T\Phi\right)\boldsymbol{a}_i + \frac{r}{q+r}\tilde{\boldsymbol{G}}_i\Phi^T\varphi\left(\boldsymbol{y}_i\right)$.

Equations (26) to (31) complete the process of one-step estimation in a kernel Kalman filter, where $\boldsymbol{a}_i$, $\tilde{\boldsymbol{P}}_i^-$, $\tilde{\boldsymbol{G}}_i$, $\tilde{\boldsymbol{P}}_i$, and $\boldsymbol{b}_i$ all can be expressed with inner product of embedded variables so that they can be calculated conveniently by kernel trick with given kernel function.

3) KKF ALGORITHM

For $\hat{\mu_i}$ in (31) which is defined in RKHS, it needs to revive the state estimation $\hat{\boldsymbol{x}_i}$ in the original input space. This kind of inverse kernel projection can be realized as the following process [34].

According to (19), it has

$$\hat{\boldsymbol{x}_i} = E\left[\boldsymbol{x}_i\right] = E\left[\boldsymbol{x}_i + \boldsymbol{v}_i\right] = E\left[\boldsymbol{y}_i\right] \tag{32}$$

with

$$\begin{aligned}E\left[\boldsymbol{y}_i\right] &= \left[E\left(\boldsymbol{y}_i^{(1)}\right), \ldots, E\left(\boldsymbol{y}_i^{(j)}\right), \ldots, E\left(\boldsymbol{y}_i^{(n_y)}\right)\right]^T\\ &= \left[E\left[\beta_1\varphi\left(\boldsymbol{y}_i\right)\right], \ldots, E\left[\beta_j\varphi\left(\boldsymbol{y}_i\right)\right], \ldots, E\left[\beta_{n_y}\varphi\left(\boldsymbol{y}_i\right)\right]\right]^T\end{aligned} \tag{33}$$

where $j \in [1, n_y]$ and n_y is the number of training samples. From (33) we may obtain

$$E\left[\beta_j\varphi\left(\boldsymbol{y}_i\right)\right] = \left\langle\beta_j, \mu_i\right\rangle \tag{34}$$

Then, $\hat{\boldsymbol{x}_i}$ can be rewritten as

$$\hat{\boldsymbol{x}_i} = \left[\left\langle\beta_1, \hat{\mu_i}\right\rangle, \ldots, \left\langle\beta_j, \hat{\mu_i}\right\rangle, \ldots, \left\langle\beta_{n_y}, \hat{\mu_i}\right\rangle\right]^T = \boldsymbol{\beta_I}^T\hat{\mu_i} \tag{35}$$

where $\beta_j(\boldsymbol{y}) = \boldsymbol{y}^{(j)}$ and $\boldsymbol{\beta_I} = \left[\beta_1, \ldots, \beta_{n_y}\right]$. $\boldsymbol{\beta_I}$ describes the inverse projection and can be estimated by least squares as

$$\hat{\boldsymbol{\beta}}_{\boldsymbol{I}}(\cdot) = \Phi\left(\Phi^T\Phi\right)^{-1}\boldsymbol{Y}^{0^T} \tag{36}$$

where $\boldsymbol{Y}^0 = \left[\boldsymbol{y}_2^0, \ldots, \boldsymbol{y}_{m+1}^0\right]$. Substituting (36) into (35) yields the approximation of the state estimation in input space, formed as

$$\begin{aligned}\hat{\boldsymbol{x}_i} &= \left[\Phi\left(\Phi^T\Phi\right)^{-1}\boldsymbol{Y}^{0^T}\right]^T\left(\Phi b_i + \frac{q}{q+r}\varphi\left(y_i\right)\right)\\ &= \boldsymbol{Y}^0\boldsymbol{b}_i + \frac{q}{q+r}\boldsymbol{y}_i\end{aligned} \tag{37}$$

Combining (26) to (31) with (37), we can get a complete iterative cycle of the kernel Kalman filter. The flowchart of the standard CEO-based KKF can be given as Algorithm 1.

IV. SLIDING WINDOW ADAPTIVE KKF FOR SYSTEM UNCERTAINTY

If the $\boldsymbol{\theta}$ of state equation (18) contains uncertain dynamics, the projected state transfer operator $\boldsymbol{F}$ as (20) will take on unknown time-variant characteristics, which is hard to be reflected by using a given training sample set. Hence, in this section, we introduce maximum correntropy criterion into the KKF to optimize the kernelized filtering gain, balancing the filter performance between optimality and robustness under the uncertain dynamics, and then design a sliding window to online update the training set, building the adaptability of the KKF algorithm to the system uncertainty.

Algorithm 1 KKF

Initialization

If $i = 0$, set

$\Upsilon = \left[\varphi\left(\boldsymbol{y}_1^0\right), \ldots, \varphi\left(\boldsymbol{y}_m^0\right)\right]$

$\Phi = \left[\varphi\left(\boldsymbol{y}_2^0\right), \ldots, \varphi\left(\boldsymbol{y}_{m+1}^0\right)\right]$

$\boldsymbol{Y^0} = \left[\boldsymbol{y}_2^0, \ldots, \boldsymbol{y}_{m+1}^0\right]$

$\boldsymbol{K} = \Upsilon^T\Upsilon,\ \boldsymbol{M} = \Phi^T\Phi, \boldsymbol{T} = \Upsilon^T\Phi$

$\boldsymbol{L} = (\boldsymbol{K} + \varsigma m\boldsymbol{I})^{-1}$

$\mu_0 = \varphi\left(\boldsymbol{y}_0\right)$

$\boldsymbol{P}_0 = \varepsilon\boldsymbol{I}$

$\boldsymbol{a}_{i+1} = \boldsymbol{L}Y^T\mu_0$

$\tilde{\boldsymbol{P}}_{i+1}^- = \lambda\boldsymbol{L}\boldsymbol{K}\boldsymbol{L}^T$

End if

Iterative Cycle

For $i = 1, \ldots,$ compute

$\tilde{\boldsymbol{G}}_i = \left[(q+r)\boldsymbol{I}_m + \tilde{\boldsymbol{P}}_i^-\boldsymbol{M}\right]^{-1}\tilde{\boldsymbol{P}}_i^-$

$\tilde{\boldsymbol{P}}_i = \frac{r}{q+r}\tilde{\boldsymbol{P}}_i^- - \frac{r}{q+r}\tilde{\boldsymbol{G}}_i\boldsymbol{M}\tilde{\boldsymbol{P}}_i^- - \frac{qr}{q+r}\tilde{\boldsymbol{G}}_i$

$\boldsymbol{b}_i = \left[\frac{r}{q+r}\boldsymbol{I}_m - \frac{r}{q+r}\tilde{\boldsymbol{G}}_i\boldsymbol{M}\right]\boldsymbol{a}_i + \frac{r}{q+r}\tilde{\boldsymbol{G}}_i\Phi^T\varphi\left(\boldsymbol{y}_i\right)$

$\hat{\boldsymbol{x}}_i = \boldsymbol{Y^0}\boldsymbol{b}_i + \frac{q}{q+r}\boldsymbol{y}_i$

$\boldsymbol{a}_{i+1} = \boldsymbol{L}Y^T\Phi\boldsymbol{b}_i + \frac{q}{q+r}\boldsymbol{L}Y^T\varphi\left(\boldsymbol{y}_i\right)$

$\tilde{\boldsymbol{P}}_{i+1}^- = \boldsymbol{L}\boldsymbol{T}\tilde{\boldsymbol{P}}_i\boldsymbol{T}^T\boldsymbol{L}^T + \frac{qr}{q+r}\boldsymbol{L}\boldsymbol{K}\boldsymbol{L}^T$

End

A. KERNEL PARAMETER OPTIMIZATION

Clearly, correntropy quantifies statistical similarity between two stochastic variables and can be described in RKHS by their correlation function. According to (12), take Gaussian kernel function as $\kappa_\sigma(x, y) = exp\left(-\frac{(x-y)^2}{2\sigma^2}\right)$ where σ is an inherent parameter showing kernel size, and the correntropy between $X = \{x_j\}$ and $Y = \{y_j\}$ can be expressed as

$$(\hat{C}(X, Y) = \frac{1}{m}\sum_{j=1}^{m}\kappa_\sigma(\mathrm{x_j}, \mathrm{y_j}) \triangleq \frac{1}{m}\sum_{j=1}^{m}\kappa_\sigma(\mathrm{x_j} - \mathrm{y_j}) \tag{38}$$

Maximizing $(\hat{C}(X, Y)$ generates

$$max\, J = \hat{C}(X, Y) = \frac{1}{m}\sum_{j=1}^{m} exp\left(-e_j^2/2\sigma^2\right) \tag{39}$$

where $e_j = x_j - y_j$.

Considering existence of stochastic uncertainties during system process and measurement, we take the price as [24]

$$J = \kappa_\sigma\left(\| \underbrace{\varphi(\boldsymbol{y_i}) - \hat{\boldsymbol{\mu}}_i}_{\boldsymbol{r_i}} \|^2_{\boldsymbol{R}_i^{-1}}\right) + \kappa_\sigma\left(\| \underbrace{\hat{\boldsymbol{\mu}}_i - \boldsymbol{F_{i-1}}\hat{\boldsymbol{\mu}}_{i-1}}_{\boldsymbol{q_i}} \|^2_{\boldsymbol{P}_{i-}^{-1}}\right) \tag{40}$$

where $\boldsymbol{R}_i^{-1}$ and $\boldsymbol{P}_{i-}^{-1}$ are weight matrices. The maximum J satisfies

$$\partial J/\partial\hat{\boldsymbol{\mu}}_i = 0 \tag{41}$$

from which the optimized system state $\hat{\boldsymbol{\mu}}_i$ in RKHS with covariance $\boldsymbol{P_i}$ can be derived, as

$$\hat{\boldsymbol{\mu}}_i = \hat{\boldsymbol{\mu}}_{i-} + \boldsymbol{G_i}(\varphi(\boldsymbol{y_i}) - \hat{\boldsymbol{\mu}}_{i-}) \tag{42}$$

$$\boldsymbol{P_i} = \left(\boldsymbol{I}_{n_i} - \boldsymbol{G_i}\right)\boldsymbol{P}_{i-} \tag{43}$$

In (42), $\boldsymbol{G_i}$ is the optimal kernel filtering gain for given σ of the Gaussian kernel and has the form as

$$\boldsymbol{G_i} = \lambda_i\boldsymbol{P}_{i-}[\lambda_i\boldsymbol{P}_{i-} + \boldsymbol{R_i}]^{-1} \tag{44}$$

$$\lambda_i = \frac{\kappa_\sigma\|\varphi(\boldsymbol{y_i}) - \hat{\boldsymbol{\mu}}_{i-1}\|^2_{\boldsymbol{R}_i^{-1}}}{\kappa_\sigma\|\hat{\boldsymbol{\mu}}_{i-1} - \hat{\boldsymbol{\mu}}_{i-}\|^2_{\boldsymbol{P}_{i-}^{-1}}} \tag{45}$$

where $\hat{\boldsymbol{\mu}}_{i-}$ and $\boldsymbol{P_{i-}}$ can be calculated iteratively by (26) and (27), respectively.

It is clear to learn, by introducing λ, a gain optimization factor, into the basic KKF as Algorithm 1, MCC establishes quantitative relationship between the inherent kernel size σ and the filtering performance, reaching the best balance of optimality and robustness under meaning of correntropy.

B. KKF-MCC ALGORITHM

Taking (48) and (49) into (26) to (31) constructs the KKF based on MCC, named KKF-MCC. Its one-step estimation can be derived by the following steps.

*Step*1: Calculate one-step prediction covariance by

$$\boldsymbol{P}_{i-} = \Phi\tilde{\boldsymbol{P}}_{i-}\Phi^{\mathrm{T}} + q\boldsymbol{I}_m \tag{46}$$

where

$$\tilde{\boldsymbol{P}}_{i-} = \boldsymbol{\Omega}\Phi\tilde{\boldsymbol{P}}_{i-}\Phi^T\boldsymbol{\Omega}^T + \frac{rq}{\lambda_{i-1}q + r}\boldsymbol{\Omega}\boldsymbol{\Omega}^T \tag{47}$$

$$\boldsymbol{\Omega} = \left(\Upsilon^T\Upsilon + \varsigma m\boldsymbol{I}_m\right)^{-1}\Upsilon^T \tag{48}$$

*Step*2: Update state estimation covariance by

$$\boldsymbol{P}_i = \frac{rq}{\lambda_i q + r}\boldsymbol{I}_m + \Phi\tilde{\boldsymbol{P}}_i\Phi^T \tag{49}$$

where

$$\tilde{\boldsymbol{P}}_i = \frac{r}{\lambda_i q + r}\tilde{\boldsymbol{P}}_{i-} - \frac{\lambda_i rq}{\lambda_i q + r}\tilde{\boldsymbol{G}}_i - \frac{\lambda_i r}{\lambda_i q + r}\tilde{\boldsymbol{G}}_i\Phi^{\mathrm{T}}\Phi\tilde{\boldsymbol{P}}_{i-} \tag{50}$$

*Step*3: Compute kernel Kalman filtering gain by

$$\boldsymbol{G}_i = \frac{\lambda_i q}{\lambda_i q + r}\boldsymbol{I}_m + \frac{\lambda_i r}{\lambda_i q + r}\Phi\tilde{\boldsymbol{G}}_i\Phi^T \tag{51}$$

where

$$\tilde{\boldsymbol{G}}_i = [(\lambda_i q + r)\,\boldsymbol{I}_m + \lambda_i\tilde{\boldsymbol{P}}_{i-}\Phi^{\mathrm{T}}\Phi]^{-1}\tilde{\boldsymbol{P}}_{i-} \tag{52}$$

*Step*4: Update state estimation by

$$\hat{\boldsymbol{\mu}}_i = \Phi\boldsymbol{s}_i + \frac{\lambda_i q}{\lambda_i q + r}\varphi(\boldsymbol{y}_i) \tag{53}$$

where

$$\boldsymbol{s}_i = \left(\frac{r}{\lambda_i q + r}\boldsymbol{I}_m - \frac{\lambda_i r}{\lambda_i q + r}\tilde{\boldsymbol{G}}_i\Phi^T\Phi\right)\boldsymbol{w}_i + \frac{\lambda_i r}{\lambda_i q + r}\tilde{\boldsymbol{G}}_i\Phi^T\varphi(\boldsymbol{y}_i) \tag{54}$$

*Step*5: Calculate one-step prediction by

$$\hat{\boldsymbol{\mu}}_{i-} = \Phi \boldsymbol{w}_{i-1} \tag{55}$$

where

$$\boldsymbol{w}_{i-1} = \left(\Upsilon^T \Upsilon + \varsigma m \boldsymbol{I}_m\right)^{-1} \Upsilon^T \left[\begin{array}{c} \dfrac{\lambda_i q}{\lambda_i q + r} \varphi(\boldsymbol{y}_{i-1}) \\ + \Phi \boldsymbol{s}_{i-1} \end{array}\right] \tag{56}$$

Equations (46), (49), (51), (53), and (55) complete one-step estimation of the KKF-MCC, where λ_i aims to control the kernel-based filtering gain optimally. Substituting (53) and (55) into (45) yields

$$\lambda_i = \kappa_\sigma(A_i)/\kappa_\sigma(B_i) \tag{57}$$

where

$$\begin{aligned} A_i &= \|\varphi(\boldsymbol{y}_i) - \hat{\boldsymbol{\mu}}_{i-1}\|^2_{\boldsymbol{R}_i^{-1}} = \frac{1}{r}[\varphi(\boldsymbol{y}_i)^{\mathrm{T}} \varphi(\boldsymbol{y}_i) \\ &\quad - \varphi(\boldsymbol{y}_i)^{\mathrm{T}} \hat{\boldsymbol{\mu}}_{i-1} - \hat{\boldsymbol{\mu}}^{\hat{T}}_{i-1} \varphi(\boldsymbol{y}_i) + \hat{\boldsymbol{\mu}}^T_{i-1} \hat{\boldsymbol{\mu}}_{i-1}] \end{aligned} \tag{58}$$

$$\begin{aligned} B_i &= \|\hat{\boldsymbol{\mu}}_{i-1} - \hat{\boldsymbol{\mu}}_{i-}\|^2_{\boldsymbol{P}_{i-}^{-1}} \\ &= \frac{1}{q} B_{1i} - \frac{1}{q} B_{2i} \left(q \tilde{\boldsymbol{P}}_{i-}^{-1} + \Phi^{\mathrm{T}} \Phi\right)^{-1} B_{2i}^T \end{aligned} \tag{59}$$

Let $\boldsymbol{e}_{i-} = \hat{\boldsymbol{\mu}}_{i-1} - \hat{\boldsymbol{\mu}}_{i-}$. It has $B_{1i} = {\boldsymbol{e}_{i-}}^T \boldsymbol{e}_{i-} = \hat{\boldsymbol{\mu}}^T_{i-} \hat{\boldsymbol{\mu}}_{i-} + \hat{\boldsymbol{\mu}}^T_{i-1} \hat{\boldsymbol{\mu}}_{i-1} - \hat{\boldsymbol{\mu}}^T_{i-} \hat{\boldsymbol{\mu}}_{i-1} - \hat{\boldsymbol{\mu}}^T_{i-1} \hat{\boldsymbol{\mu}}_{i-}$ and $B_{2i} = {\boldsymbol{e}_{i-}}^T \Phi$.

To present an entire iterative cycle of the KKF-MCC, the state estimation should be brought from RKHS back to the input space. Taking (37) into (35) and (36), we can finally obtain the estimation result of KKF-MCC in the original state space, formed as

$$\begin{aligned} \hat{\boldsymbol{x}}_i &= \boldsymbol{\beta}^{\mathrm{T}} \hat{\boldsymbol{\mu}}_i \\ &= \left[\Phi \mathbf{s}_i + \frac{\lambda_i q}{\lambda_i q + r} \varphi(\boldsymbol{y}_i)\right]^{\mathrm{T}} \Phi \left(\Phi^{\mathrm{T}} \Phi + \varsigma \boldsymbol{I}_m\right)^{-1} \boldsymbol{Y}^{0^{\mathrm{T}}} \\ &= \boldsymbol{Y}^0 \Phi^{\mathrm{T}} \Phi \left(\Phi^{\mathrm{T}} \Phi + \varsigma \boldsymbol{I}_m\right)^{-1} \boldsymbol{s}_i + \frac{\lambda_i q}{\lambda_i q + r} \boldsymbol{y}_i \end{aligned} \tag{60}$$

In Algorithm 1, stochastic disturbances are assumed as Gaussian noise in RKHS with given limited power, that is, q and r are constant during filtering estimation. It probably causes mismatch of noise distribution when system facing sudden non-Gaussian disturbances, leading to undesired estimation error. However, in the KKF-MCC algorithm, we can use λ to balance the weight between q and r. Moreover, because λ built based on kernel function contain high order statistical information from the training data in input space, robustness of the algorithm can be enhanced remarkably, particularly to outlier and heavy-tailed noises. It is worthy to notice that the KKF-MCC is an extension to Algorithm 1 that is built on the MMSE criterion. For a Gaussian kernel with $\sigma \to \infty$, there is $\lambda \to 1$, and KKF-MCC degenerates to the KKF.

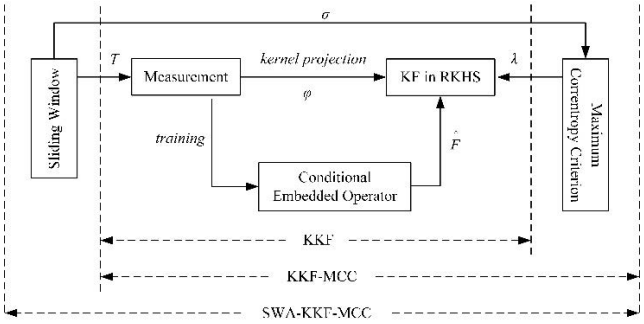

FIGURE 2. The framework of the kernel Kalman filtering algorithms.

C. SWA-KKF-MCC ALGORITHM

Built on MCC, the KKF algorithm gains robustness to noise disturbance, however it still lacks adaptability to uncertain dynamic change of system model. It is because the training samples used to construct state transfer operator for the filter has insufficient real-time information to reflect system model uncertainty, resulting in large approximation deviation of the state transfer operator. For this reason, we design a dynamic window into KKF-MCC to form a sliding window adaptive KKF-MCC, named SWA-KKF-MCC. By renewing training sample set recursively, the filter can real-time update kernel size parameter and state transfer operator. For system input samples that are periodic, state transfer operator is learnable. The framework of such adaptive kernel learning filter can be shown as Fig. 2.

Denote the training sample sets in the initial time as $\Upsilon_1 = \left[\varphi\left(\boldsymbol{y}_1^0\right), \varphi\left(\boldsymbol{y}_2^0\right), \ldots, \varphi\left(\boldsymbol{y}_m^0\right)\right]$, $\Phi_1 = \left[\varphi\left(\boldsymbol{y}_2^0\right), \varphi\left(\boldsymbol{y}_3^0\right), \ldots, \varphi\left(\boldsymbol{y}_{m+1}^0\right)\right]$, and take the average of sample distance as Gaussian kernel size parameter, i.e., $\sigma = \frac{1}{T(T-1)} \sum_{i=1}^{T} \sum_{j=1}^{T} \left|y_i^0 - y_j^0\right|, i \neq j$. Then, the state transfer operator can be online approximated by $\boldsymbol{F}_1 = \Phi_1 (\boldsymbol{K}_1 + \varsigma m \boldsymbol{I}_m)^{-1} {\Upsilon_1}^T$ where $\boldsymbol{K}_1 = {\Upsilon_1}^T \Upsilon_1$. Given the window length with T, the filter can update the training sets after T iterative cycles with

$$\Upsilon_2 = \left[\varphi\left(\boldsymbol{y}_{1+T}^0\right), \varphi\left(\boldsymbol{y}_{2+T}^0\right), \ldots, \varphi\left(\boldsymbol{y}_{m+T}^0\right)\right] \tag{61}$$

$$\Phi_2 = \left[\varphi\left(\boldsymbol{y}_{2+T}^0\right), \varphi\left(\boldsymbol{y}_{3+T}^0\right), \ldots, \varphi\left(\boldsymbol{y}_{m+1+T}^0\right)\right] \tag{62}$$

and reconfigures the state transfer operator as

$$\boldsymbol{F}_2 = \Phi_2 (\boldsymbol{K}_2 + \varsigma m \boldsymbol{I}_m)^{-1} {\Upsilon_2}^T \tag{63}$$

where $\boldsymbol{K}_2 = {\Upsilon_2}^T \Upsilon_2$.

Combining the tth sliding window and the iterative cycle of KKF-MCC, we may describe the flowchart of the SWA-KKF-MCC as Algorithm 2. In the SWA-KKF-MCC, kernel size and state transfer operator in RKHS are updated with dynamic window. For a Gaussian kernel function with $\sigma \to \infty$, Algorithm 2 degenerates to an SWA-KKF.

V. NUMERICAL SIMULATIONS

To verify the effectiveness of the kernel filtering methods, three scenarios (average sunspot prediction, hovering target

Algorithm 2 SWA-KKF-MCC

Sliding Window Update

For $t = 1, 2 \ldots$, set

Initialization

If $i = 1$, set

$$\Upsilon_t = \left[\varphi\left(\boldsymbol{y}^0_{1+(t-1)T}\right), \varphi\left(\boldsymbol{y}^0_{2+(t-1)T}\right), \ldots, \varphi\left(\boldsymbol{y}^0_{m+(t-1)T}\right)\right]$$

$$\Phi_t = \left[\varphi\left(\boldsymbol{y}^0_{2+(t-1)T}\right), \varphi\left(\boldsymbol{y}^0_{3+(t-1)T}\right), \ldots, \varphi\left(\boldsymbol{y}^0_{m+1+(t-1)T}\right)\right]$$

$$\boldsymbol{Y}^0_t = \boldsymbol{y}^0_{2+(t-1)T}, \boldsymbol{y}^0_{3+(t-1)T}, \cdots, \boldsymbol{y}^0_{m+1+(t-1)T}\Big)\Big]$$

$$\boldsymbol{K} = \Upsilon_t^T \Upsilon_t; \boldsymbol{M} = \Phi_t^T \Phi_t; \boldsymbol{T} = \Upsilon_t^T \Phi_t$$

$$\hat{\boldsymbol{\mu}}_0 = \varphi(\boldsymbol{y}_0)$$

$$\tilde{\boldsymbol{P}}_0 = \varepsilon \boldsymbol{I}_m$$

$$\boldsymbol{w_1} = \boldsymbol{L}\Upsilon^T \hat{\boldsymbol{\mu}}_0$$

$$\tilde{\boldsymbol{P}}_{1-} = \varepsilon \boldsymbol{L}\boldsymbol{K}\boldsymbol{L}^T$$

End if

The t th Iterative Cycle of KKF-MCC

For $i = 1, 2 \ldots$, set

$$\boldsymbol{Z_i} = \Phi^T \varphi(\boldsymbol{y}_i)$$

While $i = 1$

$$\lambda_i = 1$$

Else

$$A_i = \frac{1}{r}\begin{bmatrix} 1 - \varphi(\boldsymbol{y}_i)^T \hat{\boldsymbol{\mu}}_{i-1} \\ -\hat{\boldsymbol{\mu}}^T_{i-1}\varphi(\boldsymbol{y}_i) + \hat{\boldsymbol{\mu}}^T_{i-1}\hat{\boldsymbol{\mu}}_{i-1} \end{bmatrix}$$

$$B_i = \frac{1}{q}B_{1i} - \frac{1}{q}B_{2i}\left(q\tilde{\boldsymbol{P}}^{-1}_{i-} + \Phi^T\Phi\right)^{-1}B_{2i}{}^T$$

$$\lambda_i = \kappa_\sigma(A_i)/\kappa_\sigma(B_i)$$

End While

$$\tilde{\boldsymbol{G}}_i = ((\lambda_i q + r)\boldsymbol{I}_m + \lambda_i \tilde{\boldsymbol{P}}_{i-}\boldsymbol{M})^{-1}\tilde{\boldsymbol{P}}_{i-}$$

$$\tilde{\boldsymbol{P}}_i = \frac{r}{\lambda_i q + r}\tilde{\boldsymbol{P}}_{i-} - \frac{\lambda_i rq}{\lambda_i q + r}\tilde{\boldsymbol{G}}_i - \frac{\lambda_i r}{\lambda_i q + r}\tilde{\boldsymbol{G}}_i\boldsymbol{M}\tilde{\boldsymbol{P}}_{i-}$$

$$\boldsymbol{s}_i = \left(\frac{r}{\lambda_i q + r}\boldsymbol{I}_m - \frac{\lambda_i r}{\lambda_i q + r}\tilde{\boldsymbol{G}}_i\boldsymbol{M}\right)\boldsymbol{w}_i + \frac{\lambda_i r}{\lambda_i q + r}\tilde{\boldsymbol{G}}_i\boldsymbol{Z}_i$$

$$\boldsymbol{w}_{i+1} = \boldsymbol{L}\Upsilon^T\Phi\boldsymbol{s}_i + \frac{\lambda_i q}{\lambda_i q + r}\boldsymbol{L}\Upsilon^T\varphi(\boldsymbol{y}_i)$$

$$\hat{\boldsymbol{x}}_i = \boldsymbol{Y}^0_t\boldsymbol{M}(\boldsymbol{M} + \varsigma\boldsymbol{I}_m)^{-1}\boldsymbol{s}_i + \frac{\lambda_i q}{\lambda_i q + r}\boldsymbol{y}_i$$

$$\tilde{\boldsymbol{P}}_{i+1-} = \boldsymbol{L}\boldsymbol{T}\tilde{\boldsymbol{P}}_i\boldsymbol{T}^T\boldsymbol{L}^T + \frac{rq}{\lambda_i q + r}\boldsymbol{L}\boldsymbol{K}\boldsymbol{L}^T$$

End

End

tracking, and hypersonic maneuvering target tracking) are used as simulation cases, corresponding to low-, periodic-, and high-dynamic system. In the first case, we employ real data of monthly average sunspot to evaluate the prediction performance of SWA-KKF-MCC in low-dynamic system and to verify the optimization capability of using MCC. In the second, we consider ground-based radar tracking of a high-attitude flight target with hovering motion, which is to verify the effectiveness of KKF-MCC for periodic-dynamic system and the effect from inherent kernel size parameter, illustrating advantage of KKF compared with model-based Kalman filter (KF) and unscented Kalman filter (UKF). In the third case, hypersonic target tracking is used to illustrate the performance of SWA-KKF-MCC under C-type and S-type target maneuver of a hypersonic cruise vehicle (HCV), verifying effectivity of applying the kernel filtering method into non-cooperative target tracking. In the case, the target flight trajectory is generated by our developed high-fidelity simulation model integrated with CFD-based aerodynamic experiment data, satisfying necessary physical constraints such as overload, dynamic pressure and fuel consumption. We use radar for detection and non-Gaussian $\alpha-$stable noise [35] in all cases to simulate environment noise that usually has sharp impulse and heavy-tailed distribution.

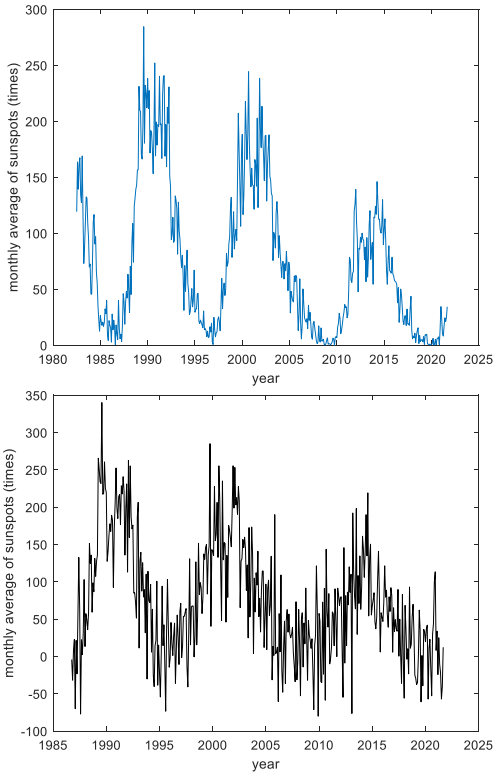

FIGURE 3. Time-series data of monthly average sunspot from May 1982 to July 2021 based on the record of world data center: true values (top) and measurement values (below).

A. PARAMETER ESTIMATION FOR LOW DYNAMIC SYSTEM

In practical observation data of average sunspot recorded by the World Data Center [36], [37], we take 471 months data from May 1982 to July 2021 as time-series true value of average sunspot and $\alpha-$stable noise, noted as $A(\alpha, \beta, \gamma, \delta)$ with probability density function

$$p(t) = \begin{cases} \exp\left(j\delta t - \gamma|t|^\alpha\left(1 - j\beta\,\mathrm{sgn}(t)\tan\left(\frac{\pi\alpha}{2}\right)\right)\right) \\ \qquad \alpha \neq 1 \\ \exp\left(j\delta t - \gamma|t|^\alpha\left(1 + j\beta\,\mathrm{sgn}(t)\frac{2}{\pi}\log|t|\right)\right) \\ \qquad \alpha = 1 \end{cases} \tag{64}$$

as measurement noise. Take $\alpha = 1.9$, representing impulse noise strength. Let $\beta = 0$, $\gamma = 3.5$, and $\delta = 0$ to generate a symmetric standard $\alpha-$stable distribution and keep the input signal-noise ratio, noted as SNR_{in}, at $6.7dB$. The true and measurement values of monthly average sunspot are shown in Fig. 3.

Use the first 50 months average sunspot as initial training samples to predict the followed 10 months average number,

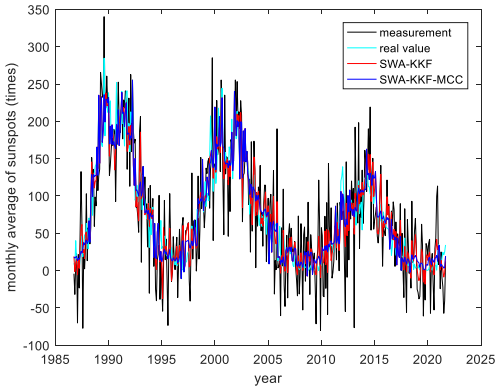

FIGURE 4. The results of monthly average sunspot prediction.

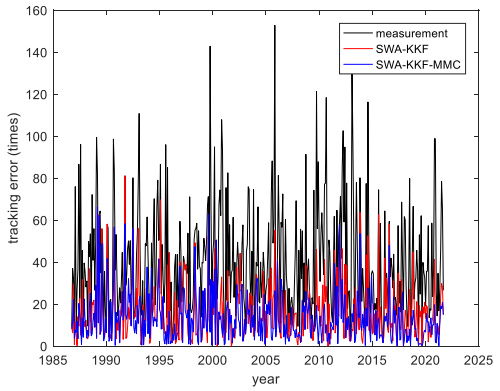

FIGURE 5. Monthly average sunspot prediction error.

TABLE 1. Mean square error comparison of average sunspot prediction.

Filter	MSE
measurement	35.7536
SWA-KKF	20.2770
SWA-KKF-MCC	17.4026

and slide window with length $T = 50$ to update the training sample set for recursive prediction. Take $\varepsilon = 0.001$, $q = 1$, and $r = 3$ in Algorithm 2, and let $\lambda = 1$ to yield an SWA-KKF for comparing with the SWA-KKF-MCC.

The prediction results of using SWA-KKF-MCC and SWA-KKF are compared as in Fig. 4 and Fig. 5. Clearly, both algorithms are effective for such low-dynamic system with $\alpha-$stable noise. Compared with SWA-KKF, the MCC-based SWA-KKF-MCC has better precision on restraining strong impulse noise contained in the measurements. Table 1 compares the mean square error (MSE) of the prediction errors shown in Fig. 5. The MSE of using SWA-KKF-MCC is decreased 14.2% than that of using SWA-KKF, indicating that MCC can help KKF to increase the adaptability to non-Gaussian noises.

B. TRAJECTORY TRACKING FOR PERIODICAL HOVERING MOTION

Besides disturbance from stochastic noise, dynamical target tracking usually has uncertain fast-changing characteristics.

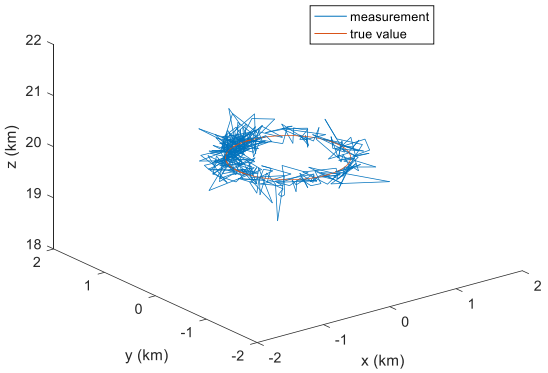

FIGURE 6. The true trajectory and measurement trajectory of hovering flight target.

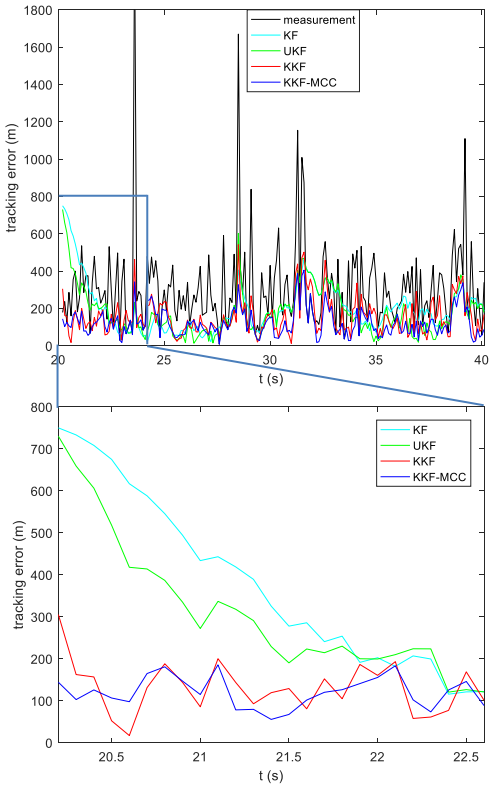

FIGURE 7. Comparison of the hovering target tracking errors of using various filtering algorithms.

In this case, we use KKF as the tracking algorithm to verify its feasibility for data-driven model-free target tracking.

Consider that a high-altitude long-endurance flight target hovers along a circle with true airspeed $V = 500km/h$ at height $20km$. Take hovering radius R as $V^2/(g\tan\eta)$ where slope angle $\eta = 11.5^\circ$ and g is gravitational acceleration. Use a ground-based radar to track the target trajectory and assume that $SNR_{in} = 6dB$ and $\alpha-$stable measurement noise takes the same values as in (64). The true and measurement target trajectories are shown in Fig. 6.

1) TRACKING PRECISION COMPARISON

Hovering maneuver yields periodic motion of the target so that the target motion can be estimated by initial training set

TABLE 2. Mean square error comparison of hovering target tracking.

Filter	MSE
measurement	305.1428
KF	200.6165
UKF	177.5543
KKF	154.8936
KKF-MCC	123.0677

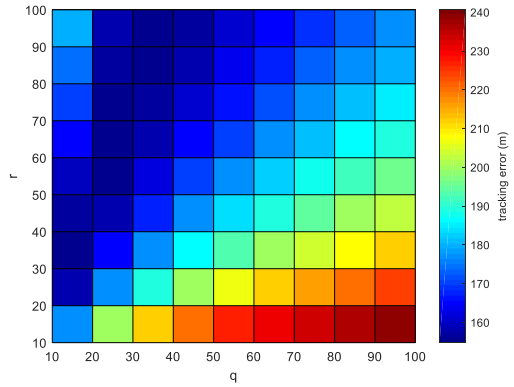

FIGURE 8. Hovering target tracking error of KKF varying with q and r.

without dynamic window. Use the KKF of Algorithm 1 and the KKF-MCC for the hovering target tracking and take the number of the initial training samples as 200 with sampling frequency 10Hz. The tracking error of these two filters that learn the state transfer operator of hovering motion and the error of the typical filters that are established on exact target motion model are compared in Fig. 7.

From the figure we can see, KKF and KKF-MCC can be independent with prior tracking model to realize motional state estimation and can achieve better tracking precision compared with KF and UKF, thanks to great tolerance of the data-driven RKHS filters to nonlinear motion and non-Gaussian noises. Table 2 gives the MSE of using various filtering algorithms, which indicates that for nonlinear non-Gaussian target tracking, KF gets the largest MSE among the compared filters and UKF reduces the tracking error through unscented transformation. Based on kernel rule and MCC, KKF-MCC gets the best precision, increasing 38.7% than KF, and establishes the robustness to stochastic noise disturbance. Furthermore, KKF-MCC gains superiority on dynamic performance of filtering estimation. As shown in the zoomed figure of Fig. 7, convergence speed of KKF and KKF-MCC is remarkably less than that of KF and UKF, implying tracking robustness to assumption deviation and system disturbance. KKF-MCC has better performance than KKF due to utilization of high order statistical information by MCC for kernel optimization.

2) KERNEL PARAMETER ANALYSIS

In KKF-MCC, λ in (57) describing system noise intensity ratio is optimizable by self-tuning q and r under MCC with

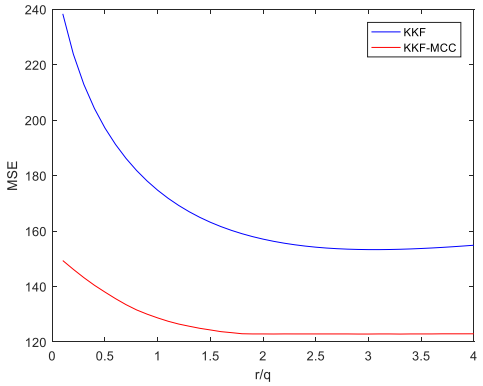

FIGURE 9. MSE comparison of hovering target tracking using KKF and KKF-MCC varying with r/q.

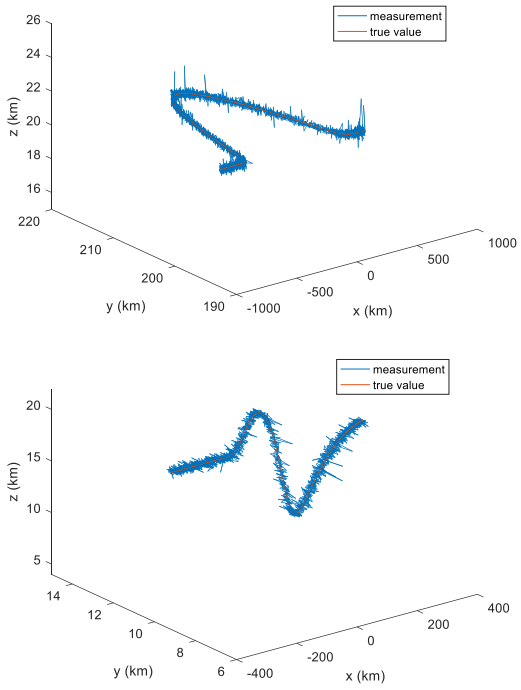

FIGURE 10. The HCV trajectories with maneuver: C-type at cruise phase (top) and S-type at descending phase (below).

given kernel size σ. However, for the basic KKF, q and r are assumed as constant, match quality of which may affect the tracking performance significantly. Fig. 8 quantifies the effect from variant q and r to tracking error of using KKF under $SNR_{in} = 6dB$. It indicates that KKF obtains the best tracking precision when $r/q = 3$ and this ratio is optimized in KKF-MCC by MCC during tracking process to balance the performance. Fig. 9 compares the tracking error MSE of KKF and KKF-MCC with variant r/q. Clearly, tracking precision of KKF-MCC is much stable than that of KKF, also because r/q in KKF-MCC has been online optimized during tracking.

C. MANEUVERING TARGET TRACKING OF HYPERSONIC VEHICLE

For nonperiodic maneuvering target tracking, we use SWA-KKF and SWA-KKF-MCC that have a sliding window

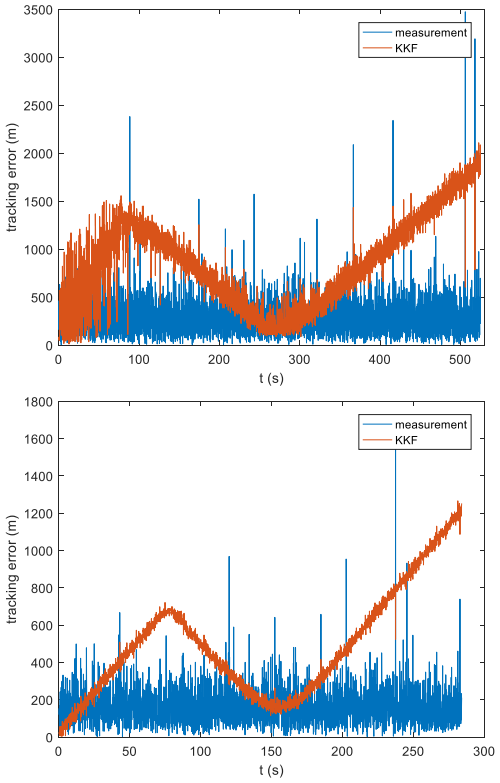

FIGURE 11. The hypersonic maneuvering target racking error using KKF: C-type (top) and S-type (below).

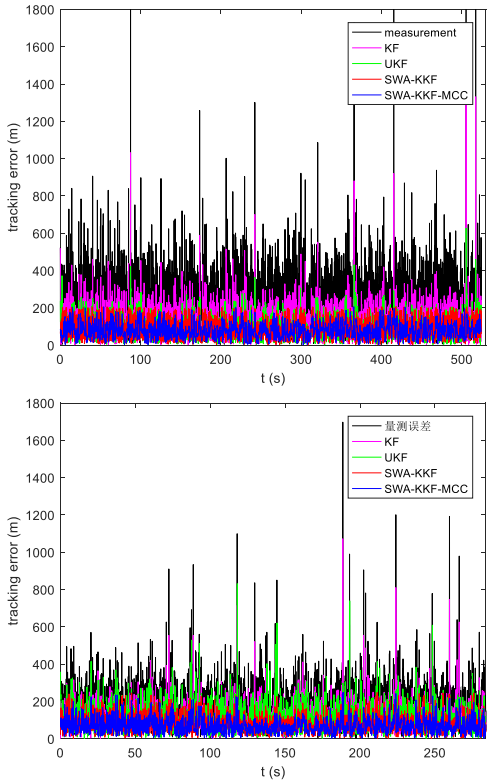

FIGURE 12. Comparison of hypersonic target tracking errors: C-type (top) and S-type (below).

to update training samples as the tracking algorithm. Consider that an HCV flying with constant speed Mach 6 and height $20km$, performing C-type maneuver in lateral plane during cruising phase and S-type maneuver in normal plane during descending phase. The HCV flight trajectory is provided by the high-fidelity simulation model in [38] with CFD-based aerodynamic data [39] and USSA76 atmospheric standard [40], optimized under multiple constraints that are overload, dynamic pressure and thrust force. On the trajectory adding an $\alpha-$stable noise with the same parameter values as in (64) gives simulated measurement. Take the initial state variance $\varepsilon = 0.001$ and $SNR_{in} = 50dB$. The true trajectory and the radar-observed trajectory with C-type and S-type maneuver can be simulated as shown in Fig. 10.

1) TRACKING PRECISION COMPARISON

To demonstrate necessity of the sliding window for kernel filtering, the basic KKF is first used as tracking algorithm with tracking error of the hypersonic maneuvering target shown in Fig. 11. The results of SWA-KKF and SWA-KKF-MCC and the model-based KF and UKF are compared in Fig. 12, with their tracking error MSE shown in Table 3. Obviously, Fig. 11 indicates that the basic KKF lacks an updating mechanism to measurement training samples and cannot adapt to the uncertain target maneuver, leading to worsened quality of learning the tracking model and raised sensitivity of tracking error to the changes of target motion. Fig. 12 shows that SWA-KKF and SWA-KKF-MCC, where the sliding window length is taken as $T = 50$, can realize precise tracking with fast convergence speed facing initial deviation and non-cooperative target maneuver, better than the tracking performance of using KF and UKF. Based on kernel optimization, SWA-KKF-MCC further improves the estimation precision. As shown in Table 3, the tracking error MSE increases 38.9% and 32.3% during C-type and S-type target maneuver, respectively, compared with KF.

TABLE 3. Mean square error comparison of the hypersonic maneuvering target tracking error.

Filter	MSE	
	C-type maneuver	S-type maneuver
measurement	231.2991	187.0762
KF	125.1878	92.2265
UKF	98.0108	84.2968
SWA-KKF	85.9351	70.5645
SWA-KKF-MCC	76.5292	62.4217

Regarding the objective of non-cooperative maneuvering target tracking is essentially the state estimation by signal denoising for non-Gaussian nonlinear dynamic system with model uncertainty, we may evaluate efficiency of tracking algorithms quantitatively with output SNR relative to input SNR. Fig. 13 shows the output SNR varying with SNR_{in}. It can be drawn from the figure that for given SNR_{in}, SWA-KKF and SWA-KKF-MCC gain larger output SNR than KF and

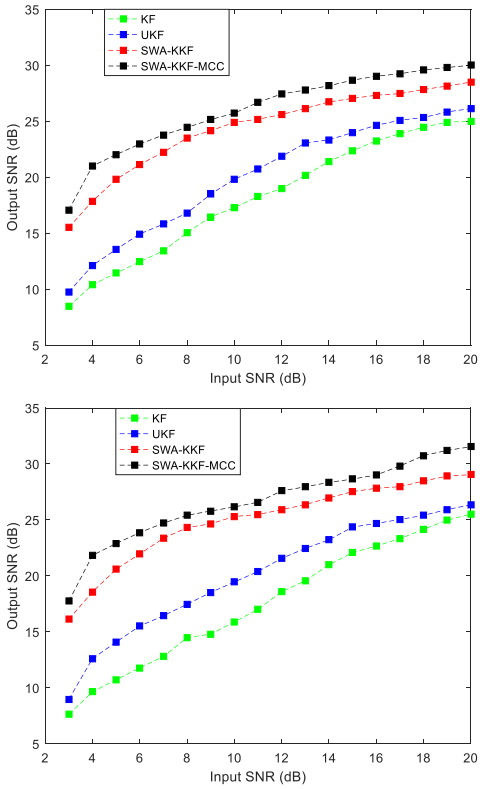

FIGURE 13. Comparison of the tracking algorithm output *SNR*: C-type (top) and S-type (below).

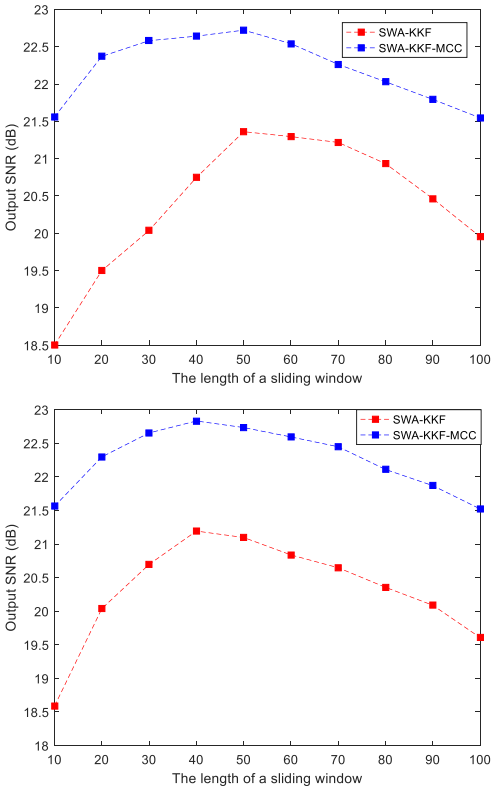

FIGURE 14. The tracking algorithm output *SNR* varying with window length: C-type (top) and S-type (below).

UKF, implying higher filter efficiency increasing with SNR_{in}. For the two windowed filters, SWA-KKF-MCC has the best tracking efficiency under the meaning of *SNR*.

2) SLIDING WINDOW ANALYSIS

The sliding window updates measurement training sample set, affecting modeling quality of state transfer operator and filtering efficiency of tracking algorithm. Fig. 14 explicates the filtering efficiency of SWA-KKF and SWA-KKF-MCC varying with the window length, where $SNR_{in} = 10$. From the figure we can see, for the case with C-type maneuver, both get the highest efficiency when the window length is 50, while for the case with S-type maneuver, it should be 40 to optimize filter efficiency. It is because higher frequency of updating the state transfer operator is required for more fierce change of the S-type maneuvering trajectory than that of C-type. Additionally, for both types of target maneuver, SWA-KKF-MCC is less sensitive to the window size than SWA-KKF. That means SWA-KKF-MCC has more stable filtering precision for various window size, implying better tracking robustness to time resolution. SWA-KKF is lack of optimization mechanism to the kernel filter gain, resulting in performance reduction. Also, Fig. 14 demonstrates that filtering efficiency decreases for both big window and small window. It is easy to know that for small window training samples are insufficient to provide enough prior knowledge of driving the filter. For big window length, importance of the early samples in window becomes weak, also leading to dropped effectiveness of prior knowledge. Particularly for a target that has uncertain maneuver, early samples in big window may cut down adaptability of the filtering gain and degenerate both tracking precision and tracking accuracy.

VI. CONCLUSION

Based on theory of nonlinear kernel projection to RKHS, in this paper, adaptive kernel learning filter SWA-KKF-MCC is proposed for solving dynamic filtering and system state estimation problem of non-Gaussian nonlinear system with model uncertainty. The filtering algorithm firstly kernelizes the Kalman filter to perform high-dimensional linearized optimal estimation in RKHS to deal with system nonlinear property, then employs a conditional embedding technique to construct learning mechanism of state transfer operator in RKHS, eliminating dependency to the prior knowledge of the system model. For non-Gaussian noise disturbance, MCC having high-order statistical information is integrated to optimize the kernel function and the kernel filtering gain, balancing filtering robustness and optimality. For uncertain dynamics of system model, a window function is designed to update the input training sample set recursively, building environmental adaptability. The simulations on time-series prediction and maneuvering target tracking have shown that the given SWA-KKF-MCC has superior filtering precision and convergence speed in low-, periodic- and high-dynamic systems with adaptability to nonlinear, non-Gaussian and

uncertain system properties and robustness to impulse noise and window length resolution.

REFERENCES

[1] W. Liu, P. P. Pokharel, and J. C. Principe, "The kernel least-mean-square algorithm," *IEEE Trans. Signal Process.*, vol. 56, no. 2, pp. 543–554, Feb. 2008.
[2] T. K. Paul and T. Ogunfunmi, "On the convergence behavior of the affine projection algorithm for adaptive filters," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 58, no. 8, pp. 1813–1826, Aug. 2011.
[3] W. Liu and J. C. Príncipe, "Kernel affine projection algorithms," *EURASIP J. Adv. Signal Process.*, vol. 2008, Mar. 2008, Art. no. 784292, doi: 10.1155/2008/784292.
[4] T. T. Gao, "Medical image analysis based on multi-target tracking," M.S. thesis, School Electron. Eng., Xidian Univ., Xi'an, China, 2012.
[5] X. Liu, C. Li, S. Li, and B. Cai, "Design and implementation of space target tracking monitoring and assistant training by simplified general performance model," in *Proc. 5th Int. Conf. Inf. Sci., Comput. Technol. Transp. (ISCTT)*, Kuala Lumpur, Malaysia, Nov. 2020, pp. 28–32.
[6] S. J. M. D. Almeida, J. C. M. Bermudez, N. J. Bershad, and M. H. Costa, "A statistical analysis of the affine projection algorithm for unity step size and autoregressive inputs," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 52, no. 7, pp. 1394–1405, Jul. 2005.
[7] L. Yu and J. Hou, "Large-screen interactive imaging system with switching federated filter method based on 3D sensor," *Complexity*, vol. 2018, pp. 1–11, Dec. 2018, doi: 10.1155/2018/8730281.
[8] M. Z. A. Bhotto and A. Antoniou, "Affine-projection-like adaptive-filtering algorithms using gradient-based step size," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 61, no. 7, pp. 2048–2056, Jul. 2014.
[9] M. Han, J. Z. Ma, W. J. Ren, and Z. Zhong, "A survey of time series online prediction based on kernel adaptive filters," *ACTA Automatica Sinica*, vol. 47, no. 4, pp. 730–746, Apr. 2021.
[10] Y. Engel, S. Mannor, and R. Meir, "The kernel recursive least-squares algorithm," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2275–2285, Aug. 2004.
[11] P. Zhu, B. Chen, and J. C. Príncipe, "A novel extended kernel recursive least squares algorithm," *Neural Netw.*, vol. 32, no. 2, pp. 349–357, Aug. 2012.
[12] Q. S. Li, Y. Zhao, L. Kou, and J. D. Wang, "An affine projection algorithm with multi-scale kernels learning," *J. Electron. Inf. Technol.*, vol. 42, no. 4, pp. 924–931, Apr. 2020.
[13] E. A. Octaria, T. Siswantining, A. Bustamam, and D. Sarwinda, "Kernel PCA and SVM-RFE based feature selection for classification of dengue microarray dataset," in *Proc. Symp. Biomath. (SYMOMATH)*, Bali, Indonesia, 2020, Art. no. 030004.
[14] R. G. Yi, "Research of online time series prediction method and application based on KRLS," M.S. thesis, Harbin Inst. Technol., Harbin, China, 2014.
[15] W. Liu, I. Park, Y. Wang, and J. C. Príncipe, "Extended kernel recursive least squares algorithm," *IEEE Trans. Signal Process.*, vol. 57, no. 10, pp. 3801–3814, Oct. 2009.
[16] Y. P. Pan, L. Xie, and H. Z. Yang, "Identification of non-uniformly sampled nonlinear systems based on KRLS," *Control Decis.*, vol. 36, no. 12, pp. 3049–3055, Dec. 2021.
[17] L. Ralaivola and F. d'Alche-Buc, "Dynamical modeling with kernels for nonlinear time series prediction," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, Whistler, CO, Canada, 2003, pp. 129–136.
[18] L. Ralaivola and F. d'Alche-Buc, "Time series filtering, smoothing and learning using the kernel Kalman filter," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Montreal, QC, Canada, 2005, pp. 1449–1454.
[19] S. Le, J. Huang, A. Smola, and K. Fukuminzu, "Hilbert space embeddings of conditional distributions with applications to dynamical systems," in *Proc. Int. Conf. Mach. Learn.*, Montreal, QC, Canada, 2009, pp. 961–968.
[20] L. Song, K. Fukumizu, and A. Gretton, "Kernel embeddings of conditional distributions: A unified kernel framework for nonparametric inference in graphical models," *IEEE Signal Process. Mag.*, vol. 30, no. 4, pp. 98–111, Jul. 2013.
[21] K. Fukumizu, L. Song, and A. Gretton, "Kernel Bayes' rule," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 24, Sep. 2011, pp. 1737–1745.
[22] G. H. W. Ebhardt, A. Kupcsik, and G. Neumann, "The kernel Kalman rule—Efficient nonparametric inference with recursive least squares," in *Proc. AAAI Conf. Artif. Intell.*, San Francisco, CA, USA, 2017, pp. 3754–3760.
[23] P. Zhu, B. Chen, and J. C. Príncipe, "Learning nonlinear generative models of time series with a Kalman filter in RKHS," *IEEE Trans. Signal Process.*, vol. 62, no. 1, pp. 141–155, Jan. 2014.
[24] L. Dang, B. Chen, S. Wang, Y. Gu, and J. C. Principe, "Kernel Kalman filtering with conditional embedding and maximum correntropy criterion," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 66, no. 11, pp. 4265–4277, Nov. 2019.
[25] S. Fakoorian, A. Santamaria-Navarro, B. T. Lopez, D. Simon, and A. A. Agha-Mohammadi, "Towards robust state estimation by boosting the maximum correntropy criterion Kalman filter with adaptive behaviors," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 5469–5476, Jul. 2021.
[26] X. X. Fan, G. Wang, and J. C. Han, "Interacting multiple model based on maximum correntropy Kalman filter," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 68, no. 8, pp. 3017–3021, Aug. 2021.
[27] K. Xiong, W. Shi, and S. Wang, "Robust multikernel maximum correntropy filters," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 67, no. 6, pp. 1159–1163, Jun. 2020.
[28] X. Liu, H. Qu, J. Zhao, and P. Yue, "Maximum correntropy square root cubature Kalman filter with application to SINS/GPS integrated systems," *ISA Trans.*, vol. 80, pp. 195–202, Sep. 2018.
[29] M. Yang, Z. H. Jiang, H. Li, H. Yang, and Q. Huang, "A novel space target-tracking method based on generalized Gaussian distribution for on-orbit maintenance robot in Tiangong-2 space laboratory," *Sci. China Technol. Sci.*, vol. 62, pp. 1045–1054, Jun. 2019.
[30] V. N. Vapnik, *Statistical Learning Theory*. Hoboken, NJ, USA: Wiley, 1998.
[31] B. Sheng, "Estimates of the norm of the Mercer kernel matrices with discrete orthogonal transforms," *Acta Math. Hungarica*, vol. 122, no. 4, pp. 339–355, Mar. 2009.
[32] K. Fukumizu, F. R. Bach, and M. I. Jordan, "Dimensionality reduction for supervised learning with reproducing kernel Hilbert spaces," *J. Mach. Learn. Res.*, vol. 5, pp. 73–99, Apr. 2004.
[33] H. V. Henderson and S. R. Searle, "On deriving the inverse of a sum of matrices," *SIAM Rev.*, vol. 23, no. 1, pp. 53–60, Jan. 1981.
[34] G. H. Bakir, A. Zien, and K. Tsuda, "Learning to find graph pre-images," in *Proc. Joint Pattern Recognit. Symp.*, in Lecture Notes in Computer Science, vol. 3175, Sep. 2004, pp. 253–261.
[35] M. J. Sun, L. Zhou, Y. J. Liu, L. J. Gu, and L. Shi, "Joint synchronization estimation of alpha stable noise frequency shift keying signal," *J. Detection Control*, vol. 43, no. 6, pp. 78–83, Dec. 2021.
[36] Y. R. Park, T. J. Murray, and C. Chen, "Predicting sun spots using a layered perceptron neural network," *IEEE Trans. Neural Netw.*, vol. 7, no. 2, pp. 501–505, Mar. 1996.
[37] A. Gkana and L. Zachilas, "Re-evaluation of predictive models in light of new data: Sunspot number version 2.0," *Sol. Phys.*, vol. 291, no. 8, pp. 2457–2472, Oct. 2016.
[38] X. L. Feng, "Research on trajectory optimization of hypersonic vehicle based on adaptive pseudo-spectrum method," M.S. thesis, School Aeronaut. Astronaut., Univ. Electron. Sci. Technol. China, Chengdu, China, 2021.
[39] X. Feng, Y. Lv, Y. Gao, and Y. Li, "Adaptive Radau pseudo-spectral optimization for descending trajectory of a hypersonic cruise vehicle," *Aerosp. Syst.*, vol. 3, no. 4, pp. 275–286, Dec. 2020.
[40] W. B. Yang, "Formulization of standard atmospheric parameters," *J. Astronaut.*, vol. 4, no. 1, pp. 83–86, Jan. 1983.

YUANKAI LI (Senior Member, IEEE) was born in 1981. He received the Ph.D. degree in control science and engineering from Shanghai Jiao Tong University, Shanghai, China, in 2011.

From 2012 to 2013, he worked as a Postdoctoral Fellow at the Department of Aerospace Engineering, Ryerson University, Toronto, ON, Canada. He is currently working as an Associate Professor with the University of Electronic Science and Technology of China, Chengdu, China. He is the author of two monographs, six book chapters, 52 academic papers, and ten inventions. His research interests include Bayesian estimation, statistical learning, target tracking, and intelligent control for aerospace unmanned systems. He is an Associate Editor of the *Aerospace Systems* journal.

JIAXIN LOU was born in 1996. She received the B.S. and M.S. degrees in electronic and communication engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2019 and 2022, respectively.

She is currently working as a Research Assistant at the Hubei Aerospace Flight Vehicle Institute. Her research interests include kernel adaptive filtering and stochastic signal processing for time series prediction and target tracking.

XIAOSU TAN was born in 1999. She received the B.S. degree in communication engineering from Southwest Minzu University, Chengdu, China, in 2021. She is currently pursuing the M.S. degree in aeronautical and astronautical science and technology with the University of Electronic Science and Technology of China.

Her research interests include collaboration and game and planning and autonomous control for multi-agent systems.

YANKE XU was born in 1985. She received the Ph.D. degree from Northwestern Polytechnical University, Xi'an, China, in 2012.

She is currently working as an Associate Research Professor with the Luoyang Optoelectro Technology Development Center, Luoyang, China. Her research interests include guidance and control, fusion, and system design for airborne vehicles.

JINPENG ZHANG was born in 1964. He received the Ph.D. degree from Beihang University, Beijing, China, in 2010.

He is currently working as a Research Professor with the Luoyang Optoelectro Technology Development Center, Luoyang, China. His research interests include guidance and control systems and overall design for advanced airborne vehicles.

ZHONGLIANG JING (Senior Member, IEEE) was born in 1960. He received the Ph.D. degree from Northwestern Polytechnical University, Xi'an, China, in 1994.

He is currently a Cheung Kong Professor with Shanghai Jiao Tong University, Shanghai, China, and the Director of the Engineering Research Center of Aerospace Science and Technology, Ministry of Education of China. His research interests include adaptive filtering, target tracking, information fusion, avionics integration, and aerospace control.