Extended Kalman Filter Using a Kernel Recursive Least Squares Observer

Pingping Zhu, Badong Chen, and José C. Príncipe

***Abstract*— In this paper, a novel methodology is proposed to solve the state estimation problem combining the extended Kalman filter (EKF) with a kernel recursive least squares (KRLS) algorithm (EKF-KRLS). The EKF algorithm estimates hidden states in the input space, while the KRLS algorithm estimates the measurement model. The algorithm works well without knowing the linear or nonlinear measurement model. We apply this algorithm to vehicle tracking, and compare the performances with traditional Kalman filter, EKF and KRLS algorithms. Results demonstrate that the performance of the EKF-KRLS algorithm outperforms these existing algorithms. Especially when nonlinear measurement functions are applied, the advantage of the EKF-KRLS algorithm is very obvious.**

I. Introduction

THE Kalman filter (KF) rooted in the state-space formulation of linear dynamical systems, provides a recursive solution to the linear optimal filtering problem, which was first given by Kalman [1]. It applies to stationary as well as to nonstationary linear dynamical environments. However, the application of the KF to nonlinear system can be difficult. Some extended algorithms have been proposed to solve this problem, such as the extended Kalman filter (EKF)[3]-[6] and the unscented Kalman filter (UKF)[6]-[9].

All these algorithms require the exact knowledge of the dynamics in order to perform filtering. Specifically, for the Kalman filter one has to know the transition matrices and measurement matrices, and the process noise and observation noise are both considered as zero mean Gaussian noise. For the EKF and UKF algorithms, one also has to know the transition functions and measurement functions. However, for many applications, these matrices, functions or assumptions about the noise cannot be obtained easily or correctly. In this paper, we still assume that the transition matrix is known, but the measurement matrix or function is assumed unknown. We learn it directly from the real data.

We construct the linear state model in the input space like the Kalman filter, while constructing the measurement model in a reproducing kernel Hilbert space (RKHS), a space of functions. We learn the measurement function in the RKHS with the estimated hidden states using the kernel recursive least squares (KRLS) algorithm. Then we use the current estimated measurement function and transition matrix to estimate the hidden states for the next step.

The organization of the paper is as follows. In section II, the traditional Kalman filter is reviewed briefly. Next the KRLS algorithm is described in section III. In section IV the EKF-KRLS algorithm is proposed, based on the Kalman filter and the KRLS algorithms. Two experiments of vehicle tracking are studied to compare this algorithm with other existing algorithms in section V. Finally, section VI gives the discussion and summarizes the conclusion and future lines of research.

Pingping Zhu, Badong Chen and José C. Príncipe are with the Department of Electrical and Computer Engineering, The University of Florida, Gainesville, USA (email: ppzhu, chenbd, principe@cnel.ufl.edu).

This work was supported by NSF award ECCS 0856441

II. Review of Kalman Filtering

The concept of state is fundamental in the Kalman filter. The state vector or simply state, denoted by $\mathbf{x}_i$, is defined as the minimal model that is sufficient to uniquely describe the unforced dynamical behavior of the system; the subscript i denotes discrete time. Typically, the state $\mathbf{x}_i$ is unobservable. To estimate it, we use a set of observed data, denoted by the vector $\mathbf{y}_i$. The model can be expressed mathematically as:

$$\mathbf{x}_{i+1} = \mathbf{F}_i \mathbf{x}_i + \mathbf{w}_i \tag{1}$$

$$\mathbf{y}_i = \mathbf{H}_i \mathbf{x}_i + \mathbf{v}_i \tag{2}$$

where $\mathbf{F}_i$ is the transition matrix taking the state $\mathbf{x}_i$ from time i to time $i+1$, and $\mathbf{H}_i$ is the measurement matrix. The process noise $\mathbf{w}_i$ is assumed to be zero-mean, additive, white, and Gaussian noise, with the covariance matrix defined by

$$E[\mathbf{w}_i \mathbf{w}_j^T] = \begin{cases} \mathbf{S}_i & for\ i = j \\ \mathbf{0} & for\ i \neq j \end{cases} \tag{3}$$

Similarly, the measurement noise $\mathbf{v}_k$ is assumed to be zero-mean, additive, white, and Gaussian noise, with the covariance matrix defined by

$$E[\mathbf{v}_i \mathbf{v}_j^T] = \begin{cases} \mathbf{R}_i & for\ i = j \\ \mathbf{0} & for\ i \neq j \end{cases} \tag{4}$$

Suppose that a measurement on a linear dynamical system, described by (1) and (2) has been made at time i. The requirement is to use the information contained in the new measurement $\mathbf{y}_i$ to update the estimation of the unknown hidden state $\mathbf{x}_i$. Let $\hat{\mathbf{x}}_i^-$ denote a priori estimate of the state, which is already available at time i. In the algorithm of Kalman filter, the hidden state $\mathbf{x}_i$ is estimated as the linear combination of $\hat{\mathbf{x}}_i^-$ and new measurement $\mathbf{y}_i$, in the form of

$$\hat{\mathbf{x}}_i = \hat{\mathbf{x}}_i^- + \mathbf{K}_i(\mathbf{y}_i - \mathbf{H}_i \hat{\mathbf{x}}_i^-) \tag{5}$$

where matrix $\mathbf{K}_i$ is called Kalman gain.

978-1-4244-9637-2/11/$26.00 ©2011 IEEE

Algorithm 1: Kalman filter
Initialization: For $i = 0$, set
$\hat{\mathbf{x}}_0 = E[\mathbf{x}_0]$
$\mathbf{P}_0 = E[(\mathbf{x}_0 - E[\mathbf{x}_0])(\mathbf{x}_0 - E[\mathbf{x}_0])^T]$
Computation: For $i = 1, 2, \ldots,$ compute:
State estimate propagation
$\hat{\mathbf{x}}_i^- = \mathbf{F}_{i-1}\hat{\mathbf{x}}_{i-1}$
Measurement prediction
$\hat{\mathbf{y}}_i = \mathbf{H}_i\hat{\mathbf{x}}_i^-$
Error covariance propagation
$\mathbf{P}_i^- = \mathbf{F}_{i-1}\mathbf{P}_{i-1}\mathbf{F}_{i-1}^T + \mathbf{S}_{i-1}$
Kalman gain matrix
$\mathbf{K}_i = \mathbf{P}_i^-\mathbf{H}_i^T\left[\mathbf{H}_i\mathbf{P}_i^-\mathbf{H}_i^T + \mathbf{R}_i\right]^{-1}$
State estimate update
$\hat{\mathbf{x}}_i = \hat{\mathbf{x}}_i^- + \mathbf{K}_i(\mathbf{y}_i - \mathbf{H}_i\hat{\mathbf{x}}_i^-)$
Error covariance update
$\mathbf{P}_i = (\mathbf{I} - \mathbf{K}_i\mathbf{H}_i)\mathbf{P}_i^-.$

The algorithm of Kalman filter is summarized in Algorithm 1. The details can be found in [2][4].

Here, $\mathbf{P}_i^-$ and $\mathbf{P}_i$ are a priori covariance matrix and a posteriori covariance matrix, respectively, defined as

$$\mathbf{P}_i^- = E[(\mathbf{x}_i - \hat{\mathbf{x}}_i^-)(\mathbf{x}_i - \hat{\mathbf{x}}_i^-)^T], \tag{6}$$

$$\mathbf{P}_i = E[(\mathbf{x}_i - \hat{\mathbf{x}}_i)(\mathbf{x}_i - \hat{\mathbf{x}}_i)^T]. \tag{7}$$

III. Kernel Recursive Least Squares Algorithm

In this section we discuss another filter algorithm, the kernel recursive least squares (KRLS) algorithm. The algorithm is a non-linear kernel-based version of the recursive least squares (RLS) algorithm proposed by Y. Engel[10]. We consider a recorded sequence of input and output samples $D_i = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_i, y_i)\}$, arising from some unknown source. In the prediction problem, one attempts to find the best predictor $\hat{y}_i$ for y_i given $D_{i-1} \cup \{\mathbf{x}_i\}$. In this context, one is often interested in on-line applications, where the predictor is updated following an arrival of each new sample. The KRLS algorithm assumes a functional form, e.g. $\hat{y}_i = f(\mathbf{x}_i)$ and minimizes the cost function J

$$J = \min_f \sum_{j=1}^{i} |y_j - f(\mathbf{x}_j)|^2 + \lambda \|f(\cdot)\|. \tag{8}$$

In the reproducing kernel Hilbert space (RKHS) denoted by $\mathcal{H}$, a space of functions, a function $f(\cdot)$ is expressed as an infinity dimensional vector in $\mathcal{H}$, denoted by $\mathbf{w} \in \mathcal{H}$, and evaluation of the function $f(\mathbf{x})$ is expressed as the inner product between $\mathbf{w}$ and $\varphi(\mathbf{x})$, where $\varphi(\cdot)$ maps $\mathbf{x}$ into $\mathcal{H}$[12]-[15],

$$f(\mathbf{x}) = \langle \mathbf{w} | \varphi(\mathbf{x}) \rangle = \mathbf{w}^T\varphi(\mathbf{x}). \tag{9}$$

So the cost function is rewritten as

$$J = \min_{\mathbf{w}} \sum_{j=1}^{i} \left|y_j - \mathbf{w}^T\varphi(\mathbf{x})\right|^2 + \lambda \|\mathbf{w}\|^2. \tag{10}$$

The KRLS algorithm solves the cost function (10) recursively and estimate $\mathbf{w}$ as a linear combination of $\{\varphi(\mathbf{x}_j)\}_{j=1}^i$,

$$\mathbf{w} = \sum_{j=1}^{i} a(j)\varphi(\mathbf{x}_j) = \Phi_i\mathbf{a}(i) \tag{11}$$

where

$$\Phi_i = [\varphi(\mathbf{x}_1), \ldots, \varphi(\mathbf{x}_i)]$$

and

$$\mathbf{a}(i) = [a(1), \ldots, a(i)]^T.$$

The KRLS algorithm is summarized in Algorithm 2. The details can be found in [10][11].

Algorithm 2: Kernel Recursive Least Squares (KRLS)
Initialize: $\mathbf{Q}(1) = (\lambda + k(\mathbf{x}_1, \mathbf{x}_1))^{-1}$, $\mathbf{a}(1) = \mathbf{Q}(1)y_1$
Iterate for $i > 1$
$\mathbf{h}(i) = [k(\mathbf{x}_i, \mathbf{x}_1), \ldots, k(\mathbf{x}_i, \mathbf{x}_{i-1})]^T$
$\mathbf{z}(i) = \mathbf{Q}(i-1)\mathbf{h}(i)$
$r(i) = \lambda + k(\mathbf{x}_i, \mathbf{x}_i) - \mathbf{z}(i)^T\mathbf{h}(i)$
$\mathbf{Q}(i) = r(i)^{-1}\begin{bmatrix} \mathbf{Q}(i-1)r(i) + \mathbf{z}(i)\mathbf{z}(i)^T & -\mathbf{z}(i) \\ -\mathbf{z}(i)^T & 1 \end{bmatrix}$
$e(i) = y_i - \mathbf{h}(i)^T\mathbf{a}(i-1)$
$\mathbf{a}(i) = \begin{bmatrix} \mathbf{a}(i-1) - \mathbf{z}(i)r(i)^{-1}e(i) \\ r(i)^{-1}e(i) \end{bmatrix}$

In this paper, y_i and $\mathbf{y}_i$ are used to denote a scalar and a $d \times 1$ vector, respectively. $\mathbf{1}_d$ denotes a $d \times 1$ vector of ones in the next section.

IV. EKF-KRLS Algorithm

Recalling the measurement model in the Kalman filter, we can extend it to a nonlinear model as

$$\mathbf{y}_i = h_i(\mathbf{x}_i) + \mathbf{v}_i. \tag{12}$$

If we map the hidden state $\mathbf{x}_k$ into a RKHS $\mathcal{H}$ which contains the function $h(\cdot)$, then (12) is expressed as

$$\mathbf{y}_i = \langle h_i, \varphi(\mathbf{x}_i) \rangle_{\mathcal{H}} + \mathbf{v}_i. \tag{13}$$

We reformulate the Kalman filter in two spaces: the hidden state model is still in the input space, while the measurement model is constructed in the RKHS $\mathcal{H}$. The whole system is reformulated as

$$\mathbf{x}_{i+1} = \mathbf{F}_i\mathbf{x}_i + \mathbf{w}_i \ (input\ space) \tag{14}$$

$$\mathbf{y}_i = \langle h_i, \varphi(\mathbf{x}_i) \rangle_{\mathcal{H}} + \mathbf{v}_i \ (\mathcal{H}\ space) \tag{15}$$

Here we assume the transition matrix $\mathbf{F}_i$ is known, but the nonlinear function $h_i(\cdot)$ is unknown. The assumptions on noise $\mathbf{w}_i$ and $\mathbf{v}_i$ are the same as mentioned in section II.

Now, we review the Kalman filter algorithm to derive the novel algorithm. Because the hidden state model is the same, the things we need to consider are the Kalman gain matrix K_i and the error covariance update equations. In light of [2], the Kalman gain is defined as

$$K_i = E\left[\mathbf{x}_{i+1}e_i^T\right] R_{e,i}^{-1} \tag{16}$$

where $e_i = \mathbf{y}_i - \hat{\mathbf{y}}_i = \mathbf{y}_i - h_i(\hat{\mathbf{x}}_i^-)$ and $R_{e,i} = E[e_ie_i^T]$. It is important to note that e_i and $e(i)$ are different. According to the orthogonality principle, we have

$$
\begin{aligned}
R_{e,i} &= E[e_i e_i^T] \\
&= E\left[\left(h_i(\mathbf{x}_i) - h_i(\hat{\mathbf{x}}_i^-) + \mathbf{v}_i\right)\right. \\
&\quad \left.\left(h_i(\mathbf{x}_i) - h_i(\hat{\mathbf{x}}_i^-) + \mathbf{v}_i\right)^T\right] \\
&= E\left[\left(h_i(\mathbf{x}_i) - h_i(\hat{\mathbf{x}}_i^-)\right)\left(h_i(\mathbf{x}_i) - h_i(\hat{\mathbf{x}}_i^-)\right)^T\right] \\
&\quad + E[\mathbf{v}_i \mathbf{v}_i^T] \\
&= E\left[\left(h_i(\mathbf{x}_i) - h_i(\hat{\mathbf{x}}_i^-)\right)\left(h_i(\mathbf{x}_i) - h_i(\hat{\mathbf{x}}_i^-)\right)^T\right] \\
&\quad + R_i
\end{aligned} \tag{17}
$$

$$
E\left[\mathbf{x}_{i+1} e_i^T\right] = \mathbf{F}_i E\left[\mathbf{x}_i e_i^T\right] + E\left[\mathbf{w}_i e_i^T\right] \tag{18}
$$

with the terms $E\left[\mathbf{x}_i e_i^T\right]$ and $E\left[\mathbf{w}_i e_i^T\right]$ given by

$$
\begin{aligned}
E\left[\mathbf{x}_i e_i^T\right] &= E\left[\left(\mathbf{x}_i - \hat{\mathbf{x}}_i^- + \hat{\mathbf{x}}_i^-\right) e_i^T\right] \\
&= E\left[\left(\mathbf{x}_i - \hat{\mathbf{x}}_i^-\right) e_i^T\right] \ since\, \hat{\mathbf{x}}_i^- \perp e_i^T \\
&= E\left[\left(\mathbf{x}_i - \hat{\mathbf{x}}_i^-\right)\left(h_i(\mathbf{x}_i) - h_i(\hat{\mathbf{x}}_i^-)\right)\right] \\
&\quad since\, \left(\mathbf{x}_i - \hat{\mathbf{x}}_i^-\right) \perp \mathbf{v}_i,
\end{aligned} \tag{19}
$$

$$
\begin{aligned}
E\left[\mathbf{w}_i e_i^T\right] &= E\left[\mathbf{w}_i \left(h_i(\mathbf{x}_i) - h_i(\hat{\mathbf{x}}_i^-) + \mathbf{v}_i\right)^T\right] \\
&= E\left[\mathbf{w}_i \mathbf{v}_i^T\right] \ since\, \mathbf{w}_i \perp \left(h_i(\mathbf{x}_i) - h_i(\hat{\mathbf{x}}_i^-)\right)^T \\
&= \mathbf{0} \qquad assume\, \mathbf{w}_i \perp \mathbf{v}_i.
\end{aligned} \tag{20}
$$

Like the error covariance $\mathbf{P}_i^- = E[(\mathbf{x}_i - \hat{\mathbf{x}}_i^-)(\mathbf{x}_i - \hat{\mathbf{x}}_i^-)^T]$ in the Kalman filter algorithm, we also need to construct $E\left[\left(\mathbf{x}_i - \hat{\mathbf{x}}_i^-\right)\left(\mathbf{x}_i - \hat{\mathbf{x}}_i^-\right)^T\right]$. We employ a first-order Taylor approximation of the nonlinear function $h_i(\cdot)$ around $\hat{\mathbf{x}}_i^-$. Specifically, $h_i(\mathbf{x}_i)$ is approximated as follows

$$
\begin{aligned}
h_i(\mathbf{x}_i) &= h_i\left(\hat{\mathbf{x}}_i^- + \left(\mathbf{x}_i - \hat{\mathbf{x}}_i^-\right)\right) \\
&\approx h_i(\hat{\mathbf{x}}_i^-) + \frac{\partial h_i}{\partial \hat{\mathbf{x}}_i^-}\left(\mathbf{x}_i - \hat{\mathbf{x}}_i^-\right).
\end{aligned} \tag{21}
$$

With the above approximate expression, we can approximate K_i by substituting from (17) to (18) into (16),

$$
K_i \approx \mathbf{P}_i^- \tilde{\mathbf{H}}_i^T \left[\tilde{\mathbf{H}}_i \mathbf{P}_i^- \tilde{\mathbf{H}}_i^T + \mathbf{R}_i\right]^{-1} \tag{22}
$$

where $\tilde{\mathbf{H}}_i = \frac{\partial h_i}{\partial \hat{\mathbf{x}}_i^-}$. The error covariance update equation is the same with the approximated Kalman gain.

Once $\tilde{\mathbf{H}}_i$ is estimated at time i, we can apply the Kalman filter algorithm to solve the prediction problem. Until now the derivation is very similar to the EKF algorithm, except that the hidden state model is not nonlinear and the function $h_i(\cdot)$ is assumed unknown. In order to obtain $\tilde{\mathbf{H}}_i = \frac{\partial h_i}{\partial \hat{\mathbf{x}}_i^-}$ we will use the KRLS algorithm mentioned in section III to estimate the function $h_i(\cdot)$ based on the predicted hidden states $\hat{\mathbf{x}}_j$ $(j \le i)$. At every time step, the EKF algorithm estimates state using the current estimated function $\hat{h}_i(\cdot)$, while the KRLS algorithm estimates the unknown function $\hat{h}_{i+1}(\cdot)$ using all available estimated hidden states. We concatenate these two algorithms to obtain a novel algorithm, EKF-KRLS algorithm, summarized in Algorithm 3.

Algorithm 3: EKF-KRLS
Initialize: For $i = 0$, set
$\hat{\mathbf{x}}_0 = E[\mathbf{x}_0]$, $\Phi_0 = [k(\hat{\mathbf{x}}_0, \cdot)]$
$\mathbf{P}_0 = E[(\mathbf{x}_0 - E[\mathbf{x}_0])(\mathbf{x}_0 - E[\mathbf{x}_0])^T]$
$Q(0) = (\lambda + k(\hat{\mathbf{x}}_0, \hat{\mathbf{x}}_0))^{-1}$, $\mathbf{a}(0) = \mathbf{1}_d^T$
Computation: For $i = 1, 2, \ldots, t$
For state filter:
$\hat{\mathbf{x}}_i^- = \mathbf{F}_{i-1}\hat{\mathbf{x}}_{i-1}$
$\hat{\mathbf{y}}_i = \mathbf{a}(i-1)^T \Phi_{i-1}^T \varphi(\hat{\mathbf{x}}_i^-)$
$\mathbf{P}_i^- = \mathbf{F}_{i-1}\mathbf{P}_{i-1}\mathbf{F}_{i-1}^T + \mathbf{S}_{i-1}$
$\tilde{\mathbf{H}}_i = \left(\frac{\partial \Phi_{i-1}}{\partial \hat{\mathbf{x}}_i^-}\mathbf{a}(i-1)\right)^T$
$\mathbf{K}_i = \mathbf{P}_i^- \tilde{\mathbf{H}}_i^T \left[\tilde{\mathbf{H}}_i \mathbf{P}_i^- \tilde{\mathbf{H}}_i^T + \mathbf{R}_i\right]^{-1}$
$\hat{\mathbf{x}}_i = \hat{\mathbf{x}}_i^- + \mathbf{K}_i\left(\mathbf{y}_i - \hat{\mathbf{y}}_i\right)$
$\mathbf{P}_i = \left(\mathbf{I} - \mathbf{K}_i \tilde{\mathbf{H}}_\mathbf{i}\right)\mathbf{P}_i^-$
For KRLS filter:
$\Phi_i = [\zeta \Phi_{i-1}, k(\hat{\mathbf{x}}_i, \cdot)]$
$\mathbf{h}(i) = \Phi_{i-1}^T \varphi(\hat{\mathbf{x}}_i)$
$\mathbf{z}(i) = \mathbf{Q}(i-1)\mathbf{h}(i)$
$r(i) = \lambda + k(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_i) - \mathbf{z}(i)^T \mathbf{h}(i)$
$\mathbf{Q}(i) = r(i)^{-1}\begin{bmatrix} \mathbf{Q}(i-1)r(i) + \mathbf{z}(i)\mathbf{z}(i)^T & -\mathbf{z}(i) \\ -\mathbf{z}(i)^T & 1 \end{bmatrix}$
$\mathbf{e}(i) = \mathbf{y}_i - \mathbf{a}(i-1)^T \mathbf{h}(i)$
$\mathbf{a}(i) = \begin{bmatrix} \mathbf{a}(i-1) - r(i)^{-1}\mathbf{z}(i)\mathbf{e}(i)^T \\ r(i)^{-1}\mathbf{e}(i)^T \end{bmatrix}$

Like the KRLS algorithm, Φ_{i-1} is scaled by a forgetting factor $\zeta (0 \ll \zeta \le 1)$ to get Φ_i.The reason is that we estimate the measurement function $h_i(\cdot)$ based on the estimated hidden states $\{\hat{\mathbf{x}}_j\}_{j=1}^i$. However, the hidden states are not trusted at the beginning of filtering. So we use the forgetting factor to get rid of the impact of wrong early estimates.

In practice, we should also consider sparsification and approximate linear dependency (ALD) [10][11] to restrict computational complexity. We can also consider discarding some beginning estimated states or just computing recent several estimated states in a running window.

V. Experiments and Results

In this paper we give two experiments to evaluate the EKF-KRLS algorithm: first, a vehicle tracking problem with linear measurement model; second, a vehicle tracking problem with nonlinear measurement model. The proposed algorithm, KF/EKF algorithm and KRLS algorithm are tested and evaluated to track the vehicle in a popular open surveillance dataset, PETS2001[16]. Since our goal is to evaluate the prediction performance of these algorithms, we mark manually the vehicle which we want to track. The figures below show the frames and the trajectory of the vehicle.

Fig. 1. Trajectory of the vehicle with background

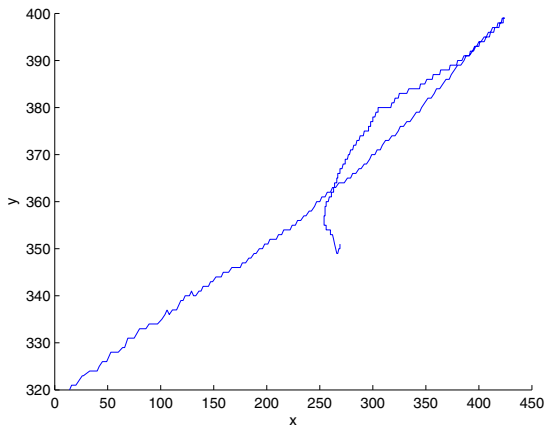

Fig. 2. Trajectory of the vehicle

In Fig. 1 the red line is the trajectory of the right front light of the vehicle. One can see that the vehicle travels straight first, then backs up, and finally parks. The trajectory is presented alone in Fig. 2. There are 410 frames in the surveillance. Considering that these algorithms have different convergent time, we compare their performances from the $50th$ frame to the end.

linear measurement model: In this experiment our observations are the vehicle positions $P(x, y)$ in a Cartesian coordinate system. **Kalman filter:** The model is

$$\mathbf{x}_{i+1} = \mathbf{F}_i \mathbf{x}_i + \mathbf{w}_i \tag{23}$$

$$\mathbf{y}_i = \mathbf{H}_i \mathbf{x}_i + \mathbf{v}_i \tag{24}$$

where $\mathbf{w}_i$ and $\mathbf{v}_i$ are process noise and observation noise with covariances $\mathbf{Q}_i$ and $\mathbf{R}_i$ respectively.

For the vehicle tracking, we choose the Ramachandra's model. Each position coordination of the moving vehicle is assumed to be described by the following equations of motion:

$$\begin{aligned} x_{i+1} &= x_i + \dot{x}_i T + \ddot{x}_i \frac{T^2}{2} \\ \dot{x}_{i+1} &= \dot{x}_i + \ddot{x}_i T \\ \ddot{x}_{i+1} &= \ddot{x}_i + a_i \end{aligned} \tag{25}$$

where x_i, $\dot{x}_i$ and $\ddot{x}_i$ are the vehicle position, the vehicle velocity and the vehicle acceleration at frame i, T is the sample time, and a_i is the plant noise that perturbs the acceleration and accounts for both maneuvers and other modeling errors. a_i is assumed to be zero mean and of constant variance σ_a^2 and also uncorrelated with its values at other time intervals.

One can find that with a larger σ_a^2 we can obtain better performance when the acceleration changes sharply, while with a smaller σ_a^2 we can obtain better performance when the acceleration changes smoothly. So for a rich position surveillance, we have to choose a proper σ_a^2 to get the best performance in the sense of the whole surveillance. We scan the parameter σ_a^2 and set $\sigma_a^2 = 110$ to obtain the best performance of the Kalman filter. Here we use the same σ_a^2 for x-coordination and y-coordination.

The linear measurement equation is written as:

$$y_i = \mathbf{H}_i x_i + v_i \tag{26}$$

where y_i is the measured position at frame i, v_i is the random noise corrupting at frame i and for each position coordination of the moving vehicle the measurement matrix $\mathbf{H}_i = [\ 1 \quad 0 \quad 0\ ]$.

The covariance of observation noise is

$$\mathbf{R}_i = r_i \mathbf{I}_2 \tag{27}$$

where $\mathbf{I}_2$ denotes the 2×2 identical matrix. We set the covariance of measurement noise $r_i = 0.1$.

KRLS: We use the KRLS algorithm to predict the next position based on the previous N positions. We choose the Gaussian kernel in the KRLS algorithm, defined as

$$k(x, y) = exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right) \tag{28}$$

where σ is called kernel size. Here the kernel size is set as 500 through trials. The forgetting factor ς is 0.85. For the 2-D vehicle tracking, actually we use $2N$ data to predict the coordinates x and y of next position respectively. We set the parameter N as 6 to obtain the best performance of the KRLS algorithm.

EKF-KRLS: For the EKF-KRLS algorithm we only need to know the transition matrix and function $h(\cdot)$ can be learned from the data. We choose the same transition matrix $\mathbf{F}_i$ as the Kalman algorithm to track the vehicle.

The EKF-KRLS algorithm learns the function $h(\cdot)$ using the KRLS algorithm with the previously estimated hidden states, and the hidden states themselves are estimated by the previous function $h(\cdot)$. The beginning estimated hidden states are not trustable. Therefore, we need a forgetting factor $0 < \varsigma < 1$ to make current hidden states more important with larger weights. We also use the running window to control the updating of the function $h(\cdot)$, which means that we learn the current $h(\cdot)$ based on the only previous m estimated hidden states. We choose these parameters as $\varsigma = 0.69$ and $m = 35$. We set the covariance of process noise as a zero matrix and the same covariance of measurement noise as the Kalman algorithm where $r_i = 0.1$. We also use the Gaussian kernel in the algorithm and the kernel size is set as 2000. These parameters are chosen to obtain the best performance.

The performances of these three algorithms are presented in Fig. 3 and the errors of the 1-step prediction are plotted in Fig. 4. The mean square errors (MSE) are summarized in TABLE I.

TABLE I
MSE FOR DIFFERENT ALGORITHMS

Algorithm	Kalman Filter	KRLS	EKF-KRLS
MSE	0.4020	0.4141	0.3742

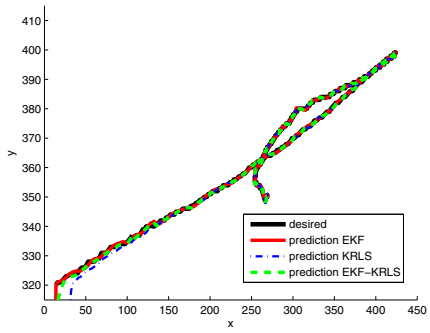

Fig. 3. Trajectories of the true position and the predictions of KF, KRLS and EKF-KRLS algorithms

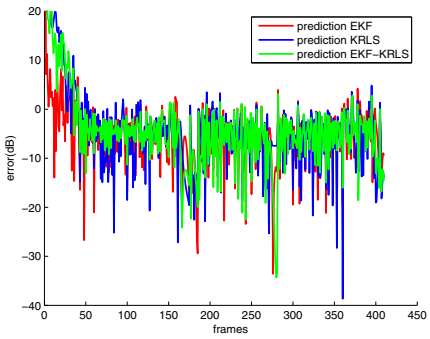

Fig. 4. Comparison of KF, KRLS and EKF-KRLS algorithms

From TABLE I, it is obvious that the EKF-KRLS algorithm has the best performance in this experiment when it uses the same transition matrix $\mathbf{F}_i$ as the Kalman filter algorithm.

In order to compare these three algorithms statistically, a Gaussian noise with a constant variance σ_n^2 is added to the original data. We run these three algorithms 100 times on the noisy data and compare the means and standard deviations of the MSE. The results are summarized in TABLE II. It shows that the EKF-KRLS algorithm has the best performances statistically for the application of vehicle tracking with a linear measurement model.

TABLE II

MSE FOR DIFFERENT ALGORITHMS WITH NOISE

	Kalman Filter	KRLS	EKF-KRLS
$\sigma_n^2 = 0.01$	0.4392 ± 0.0091	0.4378 ± 0.0068	0.4110 ± 0.0087
$\sigma_n^2 = 0.02$	0.4741 ± 0.0132	0.4627 ± 0.0111	0.4453 ± 0.0132

Nonlinear measurement model: In this experiment our observations are the distance r between the vehicle and the origin $P(0,0)$ and the slope k with respect to the origin. The distance and the slope can be expressed as

$$r = \sqrt{x^2 + y^2}, \tag{29}$$

$$k = \frac{y}{x}. \tag{30}$$

Here we use the same state model mentioned before and the nonlinear measurement functions (29) and (30). We first generate the new data from the position data using (29) and (30). Then we predict the 1-step output of the data like the first experiment. Finally, we compare the performances of these three algorithms.

EKF: Because the measurement functions are nonlinear, we use the original EKF algorithm [3] instead of the Kalman filter algorithm. The state transition matrix is the same matrix $\mathbf{F}_i$. The covariance of process noise and the covariance of measurement noise are set as $\sigma_a^2 = 240$ and $r_i = 0.1$, respectively.

KRLS: In this experiment we still use the Gaussian kernel and set the kernel size as 100. The parameter N and the forgetting fact are still 6 and 0.85, respectively.

EKF-KRLS: We choose the same transition matrix $\mathbf{F}_i$ as the EKF algorithm and set these parameters as $\varsigma = 0.64$ and $m = 35$. We set the covariance of process noise as a zero matrix and the same covariance of measurement noise $r_i = 0.1$ as the EKF algorithm. We also choose the Gaussian kernel in the algorithm and set the kernel size as 1000.

All parameters above are chosen to obtain the best performances.

Because the ranges of the distance r and the slope k are so different, we compare their performances separately. For the distance r, the trajectories and errors are plotted in Fig. 5 and Fig. 6:

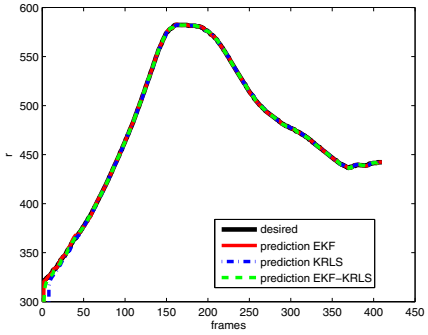

Fig. 5. Trajectories of the distance r

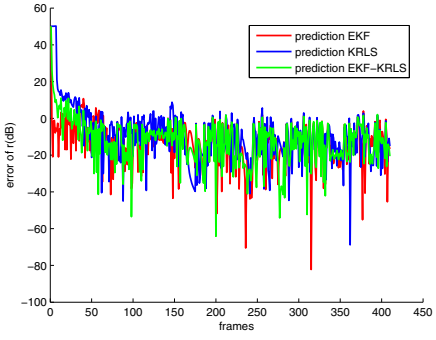

Fig. 6. Errors of the distance r

For the slope k the trajectories and errors are plotted in Fig. 7 and Fig. 8:

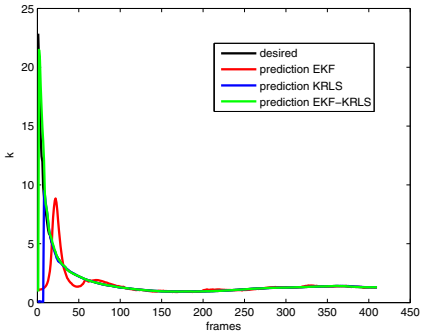

Fig. 7. Trajectories of the slope k

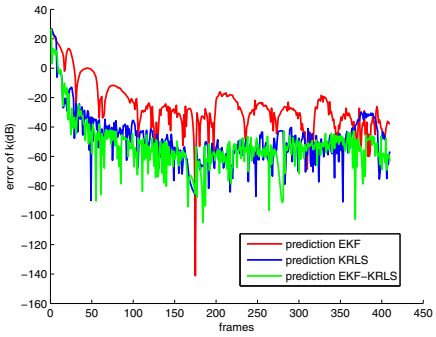

Fig. 8. Errors of the slope k

TABLE III summarizes the prediction performances of the distance r and the slope k. The results are the MSE from the $50th$ frame to the end.

TABLE III
MSE OF DISTANCE r AND SLOPE k

Algorithm	EKF	KRLS	EKF-KRLS
distance r	0.2218	0.3654	0.1715
slope k	0.0141	0.0001	0.0000

In order to compare their performances more correctly and visually, we transform the distance r and the slope k to the position $P(x, y)$ using the below equations:

$$x = \frac{r}{\sqrt{1+k^2}}, \tag{31}$$

$$y = kx. \tag{32}$$

The trajectories and errors are plotted in Fig. 9 and Fig. 10.

The TABLE IV summarizes the prediction performances of these three algorithms. The results are the MSE between predicted position and true position from the $50th$ frame to the end. One can find that the EKF-KRLS algorithm has the best tracking performance and the advantage is very obvious.

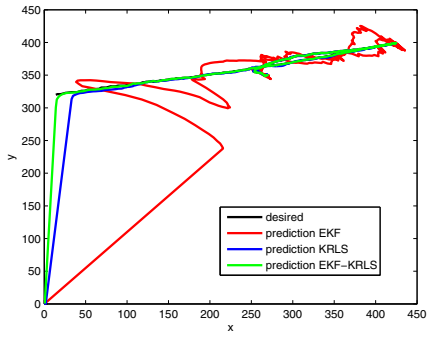

Fig. 9. Trajectories of the true position and the predictions of EKF, KRLS, and EKF-KRLS algorithms

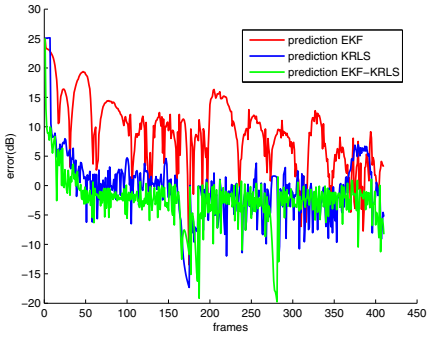

Fig. 10. Comparison of EKF, KRLS, and EKF-KRLS algorithms

VI. Discussion and Conclusion

The EKF-KRLS algorithm is presented in this paper. We construct the state model in the input space, as a linear model. The transition matrix determines the dimensionality and the dynamics of the hidden states. We construct the measurement model in the RKHS, which is a space of functions. A non-linear function $h(\cdot)$ is expressed as a linear combination of mapped hidden states. In other words, the measurement model is linear in the RKHS, but is non-linear from the input space point of view.

We connect these two models in different spaces. Like the EKF algorithm, we estimate the function $h(\cdot)$ with respect to hidden states and obtain the measurement matrix $\mathbf{H}_i$. However, our algorithm is different from the EKF algorithm. The reason is that for the EKF algorithm, the measurement function $h(\cdot)$ should be known in advance, but it is not required by the EKF-KRLS algorithm. Because the measurement function for our algorithm can be learned directly from the real data.

The goal of the EKF-KRLS algorithm is to estimate the

TABLE IV
MSE OF POSITION

Algorithm	EKF	KRLS	EKF-KRLS
MSE	10.3934	1.0401	0.5637

output. The transition matrix is the design parameter, which can be chosen to obtain the best performance. Since the measurement function $h(\cdot)$ is learned from the data, the choice of this transition matrix is very important, which reflects the dynamics of the system.

The experiments of vehicle tracking in section V show that the EKF-KRLS algorithm obtains significantly obvious advantage when the measurement functions are nonlinear. The reason is as follows. For the linear measurement model, the Kalman filter is optimal and the designed state model is very close to the real model. So the EKF-KRLS algorithm just outperforms the Kalman filer a little. For the nonlinear measurement model, the EKF is not optimal. It uses the Taylor first order expansion of the fixed nonlinear measurement function to approximate the nonlinear function. A little error in the linear state model could be amplified through the approximated nonlinear model. Although the EKF-KRLS algorithm also uses the Taylor first order expansion of the estimated nonlinear measurement function, the nonlinear function is not fixed. It is updated at every step. Therefore, the system can choose a better function to compensate the error in the state model and the measurement model. The adaptivity of the EKF-KRLS algorithm results in its obvious advantage over the other algorithms in the second experiment.

Actually, the KRLS algorithm is a special case of the EKF-KRLS algorithm. Adding the input $\mathbf{u}_i$ to the state model, we have

$$\mathbf{x}_{i+1} = \mathbf{F}_i\mathbf{x}_i + \mathbf{G}_i\mathbf{u}_i + \mathbf{w}_i \quad (33)$$

$$\mathbf{y}_i = h_i(\mathbf{x}_i) + \mathbf{v}_i. \quad (34)$$

If we consider the input $\mathbf{u}_i$ as zero-input, we get our algorithm discussed previously. If we set the matrices $\mathbf{F}_i$ and $\mathbf{G}_i$ as

$$\mathbf{F}_i = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix}_{m\times m} \quad (35)$$

$$\mathbf{G}_i = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}_{m\times 1} \quad (36)$$

the EKF-KRLS algorithm is a KRLS algorithm with processing and measurement noise. Since the state model is established in the input space, there is no change for the algorithm, but replacing $\hat{\mathbf{x}}_i^- = \mathbf{F}_{i-1}\hat{\mathbf{x}}_{i-1}$ with $\hat{\mathbf{x}}_i^- = \mathbf{F}_{i-1}\hat{\mathbf{x}}_{i-1} + \mathbf{G}_{i-1}\mathbf{u}_{i-1}$.

In this paper, we apply the EKF-KRLS algorithm to the vehicle tracking problem, which can be modeled with a linear state model and a linear measurement model. Furthermore, this algorithm can also be used to solve the problem with a linear state model and a nonlinear measurement model. Even for those problems that the measurement model is unknown, this algorithm still works. However, when the transition matrix is unknown, how to choose proper transition matrix to obtain the best performance is still an open question which needs more research in the future.

REFERENCES

[1] R. E. Kalman,"A new approach to linear filtering and prediction problem," *Transactions of the ASME, Ser. D, Journal of Basic Engineering*, 82, pp. 34-45, 1960.

[2] A. Sayed, *Fundamentals of Adaptive Filtering.* New York: Wiley, 2003.

[3] B. Anderson and J. Moore, *Optimal Filtering.* Prentice-Hall, 1979.

[4] G. Welch and G. Bishop, "An Introduction to the Kalman Filter," *Technical report, UNC-CH Computer Science Technical Report 95041,*, 1995.

[5] J. K. Uhlman, "Algorithms for multiple target tracking," *American Scientist*, vol. 80(2), pp. 128-141, 1992.

[6] S. Haykin, *Kalman Filtering and Networks.* New York: Wiley, 2001.

[7] S. J. Julier and J. K. Uhlmann, "A New Extension of the Kalman Filter to Nonlinear Systems," *Proc. of AeroSense: The 11th Int. Symp. on Aerospace/Defence Sensing, Simulation and Controls.,* 1997.

[8] E. A. Wan, Rudolph van der Merwe, and Alex T. Nelso, "Dual Estimation and the Unscented Transformation," *Advances in Neural Information Processing Systems*, no. 12, pp. 666-672, MIT Press, 2000.

[9] E. A. Wan and R. van der Merwe, "The Unscented Kalman Filter for Nonlinear Estimation," *Proc. of IEEE Symposium 2000 (AS-SPCC)*, Lake Louise, Alberta, Canada, Oct. 2000.

[10] Y. Engel, S. Mannor, and R. Meir, "The kernel recursive least-squares algorithm," *IEEE Transactions on Signal Procssing*, vol. 52, no. 8, pp. 2275-2285, Aug. 2004.

[11] W. Liu, Il Park, Y. Wang, and J. C. Principe, "Extended Kernel Recursive Least Squares Algorithm," *IEEE Transactions on Signal Procssing*, vol. 57, no. 10, pp. 3801-3814, Oct. 2009.

[12] A. Berlinet and C. Thomas-Agnan, *Reproducing Kernel Hilber Spaces in Probability and Statistics.* Kluwer Academic Publisher, 2004.

[13] N. Aronszajn, "Theory of reproducing kernels," *Trans. Amer. Math. Soc.*, vol. 68, pp.337-404, 1950

[14] T. Hofmann, B. Scholkopf, A. J. Smola, "A Review of Kernel Methods in Machine Learning," *Technical Report 156*, Max-Planck-Institut fur biologische Kybernetik, 2006

[15] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Min. Knowl. Discow.*, vol. 2, no. 2, pp. 121-167, 1998.

[16] Available: http://www.hitech-projects.com/euprojects/cantata/dataset_cantata/dataset.html