Data-driven XGBoost-based filter for target tracking

Bowen Zhai†, Wei Yi†, Ming Li†, Hao Ju†, Lingjiang Kong†
†School of Information and Communication Engineering, University of Electronic Science and Technology of China, People’s Republic of China
E-mail: kussoyi@gmail.com

Abstract: In recent years, the data-driven approach has been introduced in the field of target tracking as a powerful tool developing the end-to-end mapping relationship between input features and outputs. Typically, in data-driven methods, neural networks serve as a supplement of traditional Bayesian ﬁlters for improved estimation accuracy. However, these hybrid methods are somehow complicated to realise. In this study, inspired by the idea of direct-mapping from measurements to states, simpler method by developing a data-driven XGBoost-based Filter (DXGBF) is proposed. The DXGBF consists of four components, namely data generator, sliding window, centralisation strategy and XGBoost learner (XL). The data generator generates simulated data from the probabilistic model in the training phase. By inspecting the measurements, the sliding window enables DXGBF to track targets online. The centralisation strategy extracts the relevant kinematic information from different tracks that enables DXGBF to track randomly initialised targets. The XL is responsible for learning a function that mapping to estimate states. Simulation results show that the estimation accuracy of DXGBF is higher than those of Kalman ﬁlter, sampling importance resampling particle ﬁlter and the data-driven random-forest-based ﬁlter.

1 Introduction

Data-driven approach has become more and more popular in recent years since it is a powerful and universal tool to learn an end-to-end mapping from input features to desired outputs. It has been applied in many ﬁelds, such as medicine, transportation and security. Typically, one of the most important applications is to apply the data-driven approach to target tracking. In general, a lot of meaningful work has been done. In [1–4], neural networks (NNs) are used to predict the error between the estimate of Kalman ﬁlter (KF) [5] and the true state. The output values of NN are added to the results of KF as the ﬁnal estimation of the target state. The literatures [6, 7] study the NN-aided KF methods for GPS positioning. The innovation produced by KF is used as inputs of the NN. The NN estimates the measurement noise covariance or the gain matrix. Besides, [8, 9] propose using NN to approximate the uncertainty of the system model caused by mismodelling, extreme non-linearities etc.

Although the hybrid NN methods above can improve tracking accuracy in certain scenarios, they are complicated to realise. In [10], an idea that a learner learns a simple end-to-end mapping from the measurements to the states directly is investigated. It uses the probabilistic model of the tracking problem to simulate a large amount of state and measurement sequences. Both are fed into a random forest [11] regression algorithm that directly learns the desired mapping function. We call this method the data-driven random forest based ﬁlter (DRFF) in this paper. In the low process noise condition, DRFF shows better estimation result than Kalman smoother [12] and particle ﬁlter [13]. Although the result is impressive, DRFF has two signiﬁcant drawbacks. First, DRFF is online and to use one learner for different frames. The centralisation strategy is aimed to extract the relevant kinematic information from different tracks that enables DXGBF to track randomly initialised targets. The XL is responsible for learning a function that mapping to estimate states. Simulation results demonstrate that DXGBF is capable of online ﬁltering, and its estimation accuracy is higher than those of KF, sampling importance resampling (SIR) [15] ﬁlter and DRFF when the tracks are randomly initialised.

2 Problem formulation

We investigate a standard tracking problem in this paper. The track evolves according to a constant velocity model [16]

\[
\begin{align*}
\begin{bmatrix}
\mathbf{x}_{k+1} \\
\mathbf{y}_{k+1}
\end{bmatrix} &=
\begin{bmatrix}
1 & \Delta t & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & \Delta t \\
0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
\mathbf{x}_k \\
\mathbf{y}_k
\end{bmatrix}
+ \begin{bmatrix}
\Delta t^2/2 & 0 \\
0 & \Delta t/2 \\
0 & 0 \\
0 & 0
\end{bmatrix}
\begin{bmatrix}
\mathbf{x}_k \\
\mathbf{y}_k
\end{bmatrix}
+ \begin{bmatrix}
\mathbf{a}_k \\
\mathbf{v}_k
\end{bmatrix}
\end{align*}
\]

where \(\{\mathbf{x}_k, k \in \mathcal{N}\}\) is the state sequence, \(\{\mathbf{x}_k, \mathbf{y}_k\}\) and \(\{\mathbf{x}_k, \mathbf{y}_k\}\) are the target position and velocity in Cartesian coordinates, \(\{\mathbf{v}_k, k \in \mathcal{N}\}\) is an i.i.d process noise sequence, \(\mathbf{v}_k\) and \(\mathbf{v}_k\) are the acceleration noise in both \(x\) - and \(y\) - direction, respectively, \(\Delta t\) is the sensor scanning circle.

The sensor is located at \(r = [r_x, r_y]^T\). Its coordinate system is polar in 2D with range \(r\) and bearing \(\alpha\). The measurement model is denoted by

\[
\begin{align*}
\mathbf{z}_k &=
\begin{bmatrix}
\rho_k \\
\phi_k
\end{bmatrix} \\
&= \begin{bmatrix}
\sqrt{(\mathbf{x}_k - r_x)^2 + (\mathbf{y}_k - r_y)^2} + \mathbf{n}_{\rho,k} \\
\tan^{-1}(\mathbf{x}_k - r_x, \mathbf{y}_k - r_y) + \mathbf{n}_{\phi,k}
\end{bmatrix}
\end{align*}
\]

where \(\{\mathbf{z}_k, k \in \mathcal{N}\}\) is the measurement sequence, \(\{\mathbf{n}_k, k \in \mathcal{N}\}\) is an i.i.d measurement noise sequence, \(n_{\rho,k}\) and \(n_{\phi,k}\) are the
measurement noises in both range and bearing direction, respectively.

In particular, we seek filtered estimates of $x_k$ based on the set of all available measurements $z_{1:k} = \{ z_i, i = 1, \ldots , k \}$ up to time $k$. From the mapping perspective, the tracking problem is to design a function

$$ \mathcal{X} = g(\mathcal{X}), $$

where $\mathcal{X} = x_k$ is the output variable, $\mathcal{X}$ is the filtered state at time $k$, and $\mathcal{X} = x_{k-1}$ is the input variable. We focus on the design of this mapping function in the following section.

3 Data-driven XGBoost-based filter

In this section, a DXGBF, which can track randomly initialised target online, is developed. Fig. 1 shows the diagram of DXGBF. It contains four components, namely data generator, sliding window, centralisation strategy and XL. In the training phase, the data generator uses the probabilistic model to generate the training samples. After training, the red dotted lines are removed, and the trained DXGBF can be used to track new targets that have never been seen before. After passing the three modules ‘sliding window’, ‘centralisation strategy’, ‘XL’, the measurement sequence $z_{i,k}$ is mapped to the estimate state $\hat{x}_k$. These four components will be discussed sequentially.

3.1 Data generator

The data generator (DG) is used to generate simulated tracks in the training phase. It employs the system model (1) and the measurement model (2) to generate sufficient training samples. Suppose we generate $N$ tracks, each containing a sequence of measurements and a sequence of times from time $k = 1$ to time $k = T$. The training set can be expressed as

$$ \{(z^i_{1:k}, x^i), i = 1, \ldots , N, k = 1, \ldots , T\}, $$

which means we can obtain $NT$ training samples from the tracks. After training, the structure and the weights of the ensemble tree model of the XL are determined; hence DG can be removed from DXGBF.

3.2 Sliding window

The sliding window (SW) is to intercept the $K$ nearest measurements from the historical measurement sequence

$$ z_{k-K+1:k} = SW(z_{1:k}), $$

where $K$ is the sliding window length. The condition that $k < K$ will be discussed in Section 3.4. Compared with DRFF, which uses all the measurements to estimate states, SW uses the measurements up to the current time only, enabling online filtering. Besides, SW intercepts a fixed length measurement sequence, which allows the input of a fixed length feature vector to the XL, so it only needs one learner for all the frames.

The idea of SW is that the historical measurements received a long time ago contains little information. We can analyse the information loss of this discarding operation in a linear Gaussian system. In such a system, KF is the optimal filter, the state update formula of which being

$$ \hat{x}_k = F_k \hat{x}_{k-1} + K_k (z_k - H_k F_k \hat{x}_{k-1}) $$

(6)

where $\hat{x}_k$ is the filtered state at time $k$, $F_k$ is the state transition matrix, $H_k$ is the observation matrix, $K_k$ is the gain matrix. From (6), we can recursively derive that

$$ \hat{x}_k = w_{k-K} \hat{x}_{k-K} + w_{k-K+1} \hat{x}_{k-K+1} + \cdots + w_{k} \hat{x}_{k} $$

(7)

where $w_{k-K}$ is the weight matrix of $\hat{x}_{k-K}$ derived from all the measurements before time $k - K$, and $w_{k-K+j}$ is the weight matrix of $\hat{x}_{k-K+j}$

$$ w_{k-K} = \prod_{i=0}^{K-1} (I - K_{k-i} H_{k-i}) F_{k-i} $$

$$ w_{k-K+j} = \prod_{i=0}^{K-1} (I - K_{k-i} H_{k-i}) F_{k-i} K_{k-K+j} $$

From (7), we can see that in the linear Gaussian system, the information loss will be $w_{k-K}$ if we discard the measurements before the nearest $K$ frames. If the state transition matrix $F$, the observation matrix $H$, the process noise covariance $C$, and the measurement noise covariance $C_m$ are all known, $w_{k-K}$ can be calculated directly. With the increase of $K$’s length, the information loss will decrease. However if $K$ was too long, the input features would introduce more noise than information, so the choice of $K$ needs to be moderate.

3.3 Centralisation strategy

The centralisation strategy (CS) is to transform the measurement sequence $z_{k-K+1:k}$ to a new sequence $\tilde{z}_{k-K+1:k}$

$$ \tilde{z}_{k-K+1:k} = CS(z_{k-K+1:k}) $$

$$ = (\tilde{z}_{k-K+1, i}, \ldots , \tilde{z}_{k-1, i}), $$

(9)

$$ = (h(z_{k-K+1, i}), \ldots , h(z_{k-1, i}), h(z_{k-1, 0})), $$

where $\tilde{z}_i = h(z_{i, i}), i \neq k$ is the relative position between the ith measurement $z_i$ and the kth measurement $z_k$ in Cartesian coordinates.

![Fig. 1 Diagram of DXGBF. It contains four components: ‘data generator’, ‘sliding window’, ‘centralisation strategy’ and ‘XL’. In training phase, the ‘data generator’ is used to generate simulated measurements and states. Both are fed into DXGBF as input variables and output variables respectively. $\mathcal{Z}$ is the objective that is used for training the XL. It is a function of $x_k$ and $x_{k-1}$. After training, the red dotted lines are removed, and DXGBF can be used to track new targets that unseen before. After passing the three modules ‘sliding window’, ‘centralisation strategy’, ‘XL’, the measurement sequence $z_{1:k}$ is mapped to the estimate state $\hat{x}_k$.](image-url)
In the training phase, the processed features and bearing of the target changes. It is difficult for a learner to distinguish different obscure noise distributions, so in this paper, we only discuss the tracking problem in a constrained surveillance area, where the pdf of $\tilde{n}_k$ can be considered the same.

$3.4$ XGBoost learner

XGBoost is a tree-ensemble algorithm. In the training phase, it generates a new tree using the first- and second-order gradients of the loss function. In the prediction phase, a sample is mapped to a leaf of each tree by decision rules. The sum of the weights of all the corresponding leaves is the predict result.

The XL has two responsibilities. First, it makes up the deficiency of SW in the track initiation phase. SW requires a measurement sequence with a length of $K$. However in the track initiation phase, the length of the sequence is $<K$, which makes SW unavailable. Fortunately, XGBoost can process missing values automatically. While the length of the measurement sequence is $<K$, the missing part can be viewed as missing values, and we make up the length of the sequence to $K$.

Second, XL is a learner that learns a mapping function

$$\tilde{z}_k = XL(\tilde{z}_{K - K + 1})$$

In the training phase, the processed features $\tilde{z}_{K - K + 1}$ are fed into XL as input variables. XL tries to minimise the objective $\mathcal{Z}$, which is a function of $x_k$ and $\tilde{x}_k$. In this paper, the objective is the root mean square error (RMSE) of $x_k$ and $\tilde{x}_k$, plus the default regularisation term of XGBoost regressor. After training, the structure of the trees and the weights of the leaves of XL are determined. DXGBF can be used to track new measurements that have not been seen before. The hyper parameters of XL have a significant effect on the filtering accuracy of DXGBF, which will be discussed in the next section.

$4$ Simulation

In this section, the performance of DXGBF is investigated, compared with KF, SIR and DRFF. First, we study the effect of hyper parameters on the performance of DXGBF. Then, two scenarios of tracking fixed state and randomly initialised targets are investigated. In the following analysis, the results are gathered by averaging over $1000$ Monte Carlo realisations.

Here, the sensor is located at $r = [0, 0]^T$. The sensor scanning cycle $\Delta t = 1.0$. The sequence length of the states $T = 16.0$. The process noise and measurement noise are Gaussian distributions $v_i \sim \mathcal{N}(0, C_{v_i})$ and $n_i \sim \mathcal{N}(0, C_{n_i})$, respectively, where $C_{v_i} = \text{diag}(\sigma_{x_i}, \sigma_{y_i})$, $\sigma_{x_i} = \sigma_{y_i} = 0.05$ m/s, and

$$
C_{n_i} = \begin{bmatrix} 
\sigma_{x_i}^2 & 0 \\
0 & \sigma_{y_i}^2 
\end{bmatrix}
$$

where $\sigma_v$ and $\sigma_n$ are the standard deviation of range and bearing measurement error, respectively.

To apply KF, the measurement is converted into the Cartesian coordinates, and the measurement noise covariance in the polar coordinates is converted into Cartesian coordinates through a non-linear transformation. The number of particles of SIR is set to $3000$. Resampling is performed in each step. Both KF and SIR use the first two frame measurements to initialise a track. For DRFF, we collect all the measurements of a track and then estimate all the states offline. Both DRFF and DXGBF use $20,000$ tracks as training samples. According to the optimal result of [10], the tree numbers in each random forest is set to $150$. The sliding window length of DXGBF is $K = 13$, the number of trees is $200$, the learning rate is $0.05$, and the maximum depth of trees is $8$.

$4.1$ Effect of hyper parameters

The hyper parameters have significant influence on the filtering accuracy of DXGBF. Here we study the most important three parameters, the number of the training tracks, the number of trees of the XL, and the length of the sliding window. In Fig. 2a, the RMSE is monotonically decreasing with the increase of the number of the training tracks. When the number of the training tracks becomes very large, its performance can still be slightly improved. In Fig. 2b, the RMSE quickly converge to a constant value with the increase of the number of trees of the XL. It is noteworthy that while the trees are insufficient, the performance of DXGBF is poorly due to underfitting. In Fig. 2c, the RMSE is also tend to converge while the length of the sliding window increases. However, we should remember that the performance could be worse if $K$ was too long, because the input features would introduce more noise than information.

![Fig. 2](image-url)  

*Fig. 2 The effect of the hyper parameters*  

(a) Effect of the number of the training tracks, (b) The effect of the number of trees of XL, (c) The effect of the sliding window length  

J. Eng., 2019, Vol. 2019 Iss. 20, pp. 6683-6687

This is an open access article published by the IET under the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0/)
In this scenario, we study the tracking performance of fixed state initialised targets according to [10], for the purpose to compare with that of randomly initialised tracks. The tracks are initialised from a fixed state $[C, 50 \text{ m/s}, C, 50 \text{ m/s}]$, where $C = [C_x, C_y]$ is the initial location of all the targets. Fig. 3 shows the filtering results in different surveillance areas or measurement noise covariance conditions. The parameter configurations are shown in Table 1.

From these figures, we can see that while the tracks are generated from a fixed state, the estimate accuracy of DRFF is the best. That is because the measurements of each frame of all the tracks have small fluctuation range that it is appropriate to use the measurements as input features directly. Besides, DRFF adopts offline filtering method that it uses all the measurements to estimate the states. The performance of DXGBF is nearly the same with that of DXGBF, which demonstrate the effectiveness of the sliding window method. There is little information loss caused by the discarding operation. Moreover, DXGBF realises online filtering and only needs one learner for all the frames due to the sliding window method. The performance of both the mapping methods of DRFF and DXGBF are obviously better than the Bayesian methods in these scenarios. The upper bound of performance of the data-driven approach needs to be further studied.

### 4.2 Tracking randomly initialised targets

In this scenario, we study the tracking performance of randomly initialised targets. The initial states of the tracks are uniformly generated in the interval $[C_x \pm 500 \text{ m}, C_y \pm 500 \text{ m}]$, where $C = [C_x, C_y]$ is the centre of the surveillance area. Fig. 4 shows the filtering results in different surveillance areas or measurement noise covariance conditions. The parameter configurations of the centres of the surveillance area and the measurement noise covariance are the same as that in Section 4.2.

From these figures, we can see that the performance of DRFF decreases dramatically. That is because when the tracks are randomly initialised, the fluctuation of the measurements is in a large range. The method that directly uses the raw measurements as inputs is not appropriate any more. Interestingly, the shape of the RMSE of DRFF is like a bowl. That is because the more closer to the centre, the more information can be inferred from the forward and backward measurements.

The filtering results of KF and SIR are nearly unchanged compared with that in Fig. 3. The reason is that the tracking performance of these Bayesian methods is not affected by the initial states of the targets.

The filtering accuracy of DXGBF is also decreased due to the random initialisation of the tracks. However even though its RMSE is the lowest among these filters. It demonstrates the effectiveness of the centralisation method, which extract the relative movement information among different tracks. The results of Figs. 3 and 4 show that the simple mapping methods based on the simulated data, which is generated from the models, can achieve better filtering accuracy than the model-based Bayesian methods in certain scenarios. Nevertheless it is difficult to give a theoretically proof that why the performance of the mapping method is better than that of Bayesian methods in these scenarios. The upper bound of performance of the data-driven approach needs to be further studied.

### 4.3 Tracking randomly initialised targets

In this scenario, we study the tracking performance of randomly initialised targets. The initial states of the tracks are uniformly generated in the interval $[C_x \pm 500 \text{ m}, C_y \pm 500 \text{ m}], where C = [C_x, C_y]$ is the centre of the surveillance area. Fig. 4 shows the filtering results in different surveillance areas or measurement noise covariance conditions. The parameter configurations of the centres of the surveillance area and the measurement noise covariance are the same as that in Section 4.2.

From these figures, we can see that the performance of DRFF decreases dramatically. That is because when the tracks are randomly initialised, the fluctuation of the measurements is in a large range. The method that directly uses the raw measurements as inputs is not appropriate any more. Interestingly, the shape of the RMSE of DRFF is like a bowl. That is because the more closer to the centre, the more information can be inferred from the forward and backward measurements.

The filtering results of KF and SIR are nearly unchanged compared with that in Fig. 3. The reason is that the tracking performance of these Bayesian methods is not affected by the initial states of the targets.

The filtering accuracy of DXGBF is also decreased due to the random initialisation of the tracks. However even though its RMSE is the lowest among these filters. It demonstrates the effectiveness of the centralisation method, which extract the relative movement information among different tracks. The results of Figs. 3 and 4 show that the simple mapping methods based on the simulated data, which is generated from the models, can achieve better filtering accuracy than the model-based Bayesian methods in certain scenarios. Nevertheless it is difficult to give a theoretically proof that why the performance of the mapping method is better than that of Bayesian methods in these scenarios. The upper bound of performance of the data-driven approach needs to be further studied.

### 5 Conclusion

In this paper, we considered the target tracking problem for the data-driven approach. Specifically, a data-driven XGBoost-based filter, which is a directly mapping from measurements to states is developed. It consists of four components, namely data generator, sliding window, centralisation strategy and XL. The data generator is used for generating simulated data from the probabilistic model in the training phase. By intercepting the measurements, the sliding window enables DXGBF to track targets online and use one learner for different frames. The centralisation strategy extracts the relevant kinematic information from different tracks that enables DXGBF to track randomly initialised targets. In addition to processing missing values, the XL is responsible for learning a function that maps from processed features to estimate states. Simulation results showed that DXGBF has the ability of online filtering, and its estimation accuracy is better than those of KF, SIR and DRFF while the tracks are randomly initialised.

### 6 Acknowledgements

This work was supported in part by the Chang Jiang Scholars Program, in part by the 111 Project No. B17008, in part by the National Natural Science Foundation of China under Grant [http://creativecommons.org/licenses/by/3.0/]
References

[1] Chin, L.: 'Application of neural networks in target tracking data fusion', IEEE Trans. Aerosp. Electron. Syst., 1994, 30, (1), pp. 217–228

[2] Vaidehi, V., Chitra, N., Chokkalingam, M., et al.: 'Neural network aided Kalman filtering for multitarget tracking applications', Comput. Electr. Eng., 2001, 27, (2), pp. 217–228

[3] Sun, M., Ma, Z., Li, Y.: ‘Maneuvering target tracking using IMM Kalman filter aided by Elman neural network’. Int. Conf. on Intelligent Human-Machine Systems and Cybernetics, Hangzhou, People's Republic of China, August 2015, vol. 1, pp. 144–148

[4] Zhu, Z., Liu, W., Dai, X.: ‘Maneuvering target tracking method based on RBF neural network and interacting multi models’. J. Missiles Guidance, 2006, 26, pp. 329–333

[5] Kalman, R.E.: ‘A new approach to linear filtering and prediction problems’, J. Basic Eng., 1960, 39, (1), pp. 35–45

[6] Iwo, D.J., Huang, H.C.: ‘Neural network aided adaptive extended Kalman filtering approach for DGPS positioning’, J. Navig., 2004, 57, (3), pp. 449–463

[7] Guo, H.: ‘Neural network aided Kalman filtering for integrated GPS/INS navigation system’, Telkomnika Indonesian J. Electr. Eng., 2013, 11, (3), pp. 1221–1226

[8] Zhan, R., Wan, J.: ‘Neural network-aided adaptive unscented Kalman filter for nonlinear state estimation’, IEEE Signal Process. Lett., 2006, 13, (7), pp. 445–448

[9] Miao, Z., Shi, H., Zhang, Y., et al.: ‘Neural network-aided variational Bayesian adaptive cubature Kalman filtering for nonlinear state estimation’, Meas. Sci. Technol., 2017, 28, p. 105003

[10] Thormann, K., Siggges, F., Baum, M.: ‘Learning an object tracker with a random forest and simulated measurements’, Int. Conf. on Information Fusion, Xi'an, People's Republic of China, July 2017, pp. 1–4

[11] Breiman, L.: ‘Random forests’. Mach. Learn., 2001, 45, (1), pp. 5–32

[12] Chan, W.L., Hsiao, F.B.: ‘Implementation of the Rauch-Tung-Striebel smoother for sensor compatibility correction of a fixed-wing unmanned air vehicle’, Sensors, 2011, 11, (14), pp. 3738–3764

[13] Andampalam, M.S.: ‘A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking’, IEEE Trans. Signal Process., 2002, 50, (2), pp. 174–188

[14] Chen, T., Guestrin, C.: ‘XGBoost: a scalable tree boosting system’. ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 2016, pp. 785–794

[15] Gordon, N., Salmond, D., Smith, A.F.M.: ‘Novel approach to nonlinear and non-Gaussian Bayesian state estimation’, IEE Proc. F (Radar Signal Process.), 1993, 140, pp. 107–113

[16] Bar-Shalom, Y., Li, X.R., Kirubarajan, T.: ‘Estimation with applications to tracking and navigation: theory, algorithms and software’ (Wiley, New York, NY, USA, 2001)