Correlative Mean-Field Filter for Sequential and Spatial Data Processing

Jian Gao and Hamidou Tembine
Computer Eng. Dept., New York University Abu Dhabi, UAE
Email: {jg4631, tembine}@nyu.edu

***Abstract*—Accurate and robust state estimation is a fundamental problem in signal processing. Particle filter is an effective tool to solve the filtering problem in nonlinear stochastic dynamic systems. However, when the system is mean-field dependent and the data is high-dimensional in spatial and temporal domain, the estimator may become inaccurate or even diverge. In this paper, we propose a Correlative Mean-Field (CMF) filter for a general class of nonlinear systems. The algorithm iterates in four stages: decomposition, sampling, prediction, and correction. An expectation term is incorporated into system transition model to capture the mean-field property of the sequential data. By exploring the property of the circulant matrix and its relationship with Fast Fourier Transform (FFT), sufficient virtual samples are efficiently generated by cyclic shifts of original samples in the spacial domain. The correction is modeled as an online learning problem where the sample weights are updated by the correlation output of a regression function. Optimal states are estimated by the weighted sum. We perform simulations to illustrate that under some conditions our estimator converges while traditional mean-field-free filters diverge. Finally, we implement CMF in vehicle tracking tasks and tested on 12 traffic video sequences. Experiment results show that CMF outperforms the existing mean-field-free filters.**

***Keywords*—vehicle tracking, mean-field filter, circulant matrix, fourier transform**

I. Introduction

Given noisy observation of a signal, many problems involve estimating real system states to extract meaningful information. In Bayesian approaches [1], the solution is defined to be the posterior probability distribution of the system states given its observation history. It is a complete solution for the state estimation as it reflects the entire uncertainty, and the optimal state is estimated by the expectation of the distribution [2].

With assumptions of linearity, Gaussianity or finite state space, analytical approaches such as Kalman filter [3] and hidden Markov models [4] can achieve closed-form solutions. However, those assumptions are not valid in many realistic tasks. Particle filter (PF) [5], as a powerful nonlinear estimator, gives an alternative to the realistic situations without those assumptions. In each iteration, the posterior is approximated by a large number of random samples. Those samples are propagated over time based on the system dynamic. When new observations come, particle weights are refined by the likelihood function. Optimal state is estimated by the particle with highest weight. Despite the great success achieved, particle filter is still risky in some cases [6]. A dynamical system is mean-field dependent [7] if it involves probability measure of the state variable in the transition kernel to the next state.

(a) -30,+30 (b) -15,+15 (c) Original (d) +15,-15 (e) +30,-30

Fig. 1: Cyclic shifts of an original sample

In this situation, mean-field free approaches like PF, EnKF [8] and PSO [9] may fail, which will be shown in our simulations.

The contributions of this paper are summarized as follows:

1) Based on our previous work [10], we propose a Correlative Mean-Field (CMF) filtering framework, which recursively estimates the filtering distribution in four stages: decomposition, sampling, prediction and correction.

2) We generate sufficient virtual samples by cyclic shifts of the original sample. By exploring the properties of circulant matrix, sampling and correction are efficiently computed in Fourier domain, which is a novelty and improvement over [11].

3) We model correction stage as a learning problem. Sample weights are updated by the response of a regression function.

4) We test our algorithm in both simulations and real experiments on video sequences. Experimental results demonstrated the advantage of our algorithm in comparison with the existing mean-field free filters.

This paper is organized as follows. Section II introduces the background on circulant matrix. Section III presents the CMF filtering framework in detail. Both simulation and real experiments are performed in Section IV. Finally, the paper is concluded in Section V.

II. Background on Circulant Matrix

Circulant Matrix [12] is a prevalent mathematics tool that connects signal processing with machine learning algorithms. It explores the circulant structure of the signal in space domain and enables the usage of Fast Fourier Transform (FFT) to exhaustively extract virtual training samples by cyclic shift from an original sample. In particle filtering framework, sampling and correction are allowed to be performed in Fourier domain, which makes the computational cost independent of the sample size. Suppose the original sample is represented by a vector

978-1-5090-3843-5/17/$31.00

$x \in \mathbb{R}^m$, its corresponding circulant matrix $\mathcal{C}(x) \in \mathbb{R}^{m\times m}$ is

$$\mathcal{C}(x) = \begin{pmatrix} x_1 & x_2 & \dots & x_m \\ x_m & x_1 & \dots & x_{m-1} \\ \vdots & \vdots & \ddots & \vdots \\ x_2 & x_3 & \dots & x_1 \end{pmatrix} \tag{1}$$

Define the cyclic shift operator $T : \mathbb{R}^m \to \mathbb{R}^m$ by

$$T(x_1, x_2, \dots, x_m) = (x_m, x_1, \dots, x_{m-1}) \tag{2}$$

The j^{th} row of $\mathcal{C}(x)$ is $T^{j-1}x$. Since cyclic shift operator can model any signal translation in space domain [13], the circulant matrix $\mathcal{C}(x)$ densely represents all translated versions of the original sample x. Figure 1 shows some generated virtual samples based on an original sample.

Consider two vectors $x, x' \in \mathbb{R}^m$, with fourier transform $\mathbf{x} = \mathcal{F}(x), \mathbf{x'} = \mathcal{F}(x')$, $\mathbf{x}, \mathbf{x'} \in \mathbb{C}^m$. The circulant matrices $\mathcal{C}(x), \mathcal{C}(x')$ have the following properties [14]

$$\begin{aligned}
&\mathcal{C}(x)\mathcal{C}(x') = \mathcal{C}(x')\mathcal{C}(x) && (commutation)\\
&\alpha\mathcal{C}(x) + \beta\mathcal{C}(x') = \mathcal{C}(\alpha x + \beta x') && (linearity)\\
&\mathcal{C}^T(x)x' = \mathcal{F}^{-1}(\mathbf{x} \circ \mathbf{x'}) && (correlation)\\
&\mathcal{C}(x)\mathcal{C}(x') = \mathcal{C}(\mathcal{F}^{-1}(\mathbf{x} \circ \mathbf{x'})) && (matrix\ product)\\
&\mathcal{C}^{-1}(x) = \mathcal{C}(\mathcal{F}^{-1}(\mathbf{x}^{-1})) && (matrix\ inverse)\\
&\mathcal{C}^T(x) = \mathcal{C}(\mathcal{F}^{-1}(\mathbf{x}^{\star})) && (matrix\ transpose)
\end{aligned} \tag{3}$$

where $\circ$ denotes the Hadamard product and $\mathbf{x}^\star$ is the complex conjugation of $\mathbf{x}$. According to the Convolution Theorem [12], circular convolution of two vectors x, x' satisfies

$$x * x' = \mathcal{F}^{-1}(\mathbf{x} \circ \mathbf{x'}) = \mathcal{C}^T(x)x' \tag{4}$$

III. Proposed Scheme

A. Mean-Field-Dependent Model

A mean-field-dependent system is a dynamical system that involves the probability measure of the state variable in the transition kernel. Consider a process $(x_t)_t$ with the following transition probability

$$\mathbb{P}(x_{t+1} \in \mathcal{X} | x_t, \mathcal{L}_{x_t}) \tag{5}$$

the next state x_{t+1} depends on not only the current state x_t, but also its probability distribution $\mathcal{L}_{x_t}$. Suppose the process $(y_t)_t$ denotes the observation with conditional probability $\mathbb{P}(y_t \in \mathcal{Y} | x_t, \mathcal{L}_{x_t})$, our goal is to estimate the state variable x_t based on the observation history $y_1, \dots, y_t$. To solve this problem, we introduce the posterior filtering distribution

$$m_t(\mathcal{X}) = \mathbb{P}(x_t \in \mathcal{X} | y_1, \dots, y_t) \tag{6}$$

Once m_t is computed, we get a complete representation of the uncertainty. Assume the filter $m_t(\mathcal{X})$ has Mckean-Vlasov property, that is, the current state distribution m_t depends only on the last one m_{t-1} and the current observation y_t. This is crucial when the data sequence is high-dimensional, because the filter is independent of the entire observation history $y_1, \dots, y_{t-1}$, and thus it is suitable for online implementation. The system states can be updated recursively with consecutively incoming observations.

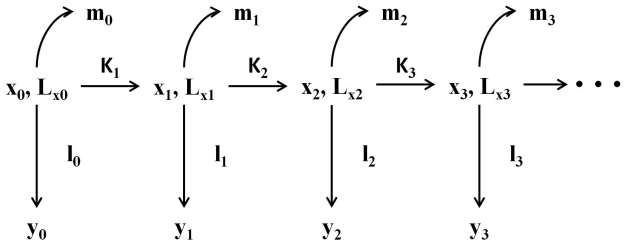


Fig. 2: Recursive State Generation

1) Prediction and Recursive State Generation: The evolution of probability distribution of the state x_t is

$$\mathcal{L}_{t+1}(\mathcal{X}) = \int_{x\in\mathcal{X}} \mathbb{P}(x_{t+1} \in \mathcal{X} \mid x, \mathcal{L}_{x_t}) \mathcal{L}_{x_t}(dx) \tag{7}$$

Then the state distribution from time step 0 to t can be generated recursively given the initial state distribution and the transition probability.

$$\begin{aligned}
&\mathbb{P}((x_{t'})_{0\le t'\le t} \in \prod_{t'\le t} \mathcal{X}_{t'}) = \\
&\mathbb{P}(x_0 \in \mathcal{X}_0) \prod_{t'=0}^{t-1} \mathbb{P}(x_{t'+1} \in \mathcal{X}_{t'+1} | x_{t'}, \mathcal{L}_{x_{t'}})
\end{aligned} \tag{8}$$

The joint probability of state-observation (x_t, y_t) evolves as

$$\begin{aligned}
&\mathbb{P}((x_t, y_t) \in \mathcal{X}\times\mathcal{Y} | x_{t-1}, \mathcal{L}_{x_{t-1}}, y_{t-1}) = \\
&\int_{(x_t,y_t)\in\mathcal{X}\times\mathcal{Y}} \mathcal{K}_{t-1}(x_{t-1}, \mathcal{L}_{x_{t-1}}; x_t) l_t(x_t, \mathcal{L}_{x_t}; y_t) \\
&\qquad * \mu_{x_t}(dx_t)\nu_{y_t}(dy_t)
\end{aligned} \tag{9}$$

where $\mathcal{K}_t$ is the mean-field state transition kernel at time step t, l_t is the likelihood of state x and its distribution $\mathcal{L}_x$ given observations y, μ_{x_t} and ν_{y_t} are distributions of signal and observation noise. Given the initial distribution, system states are generated recursively based on these probability distributions. The procedure is shown in Figure 2.

The filtering distribution m_t solves the following relations:

$$\langle \phi_t, m_t \rangle = \frac{1}{\bar{m}_t} \int m_0(dx_0) \prod_{t'<t} (l_{t'}\mathcal{K}_{t'}) \phi(x_{t'}) \tag{10}$$

$$\begin{aligned}
&\bar{m}_t = \langle 1, m_t \rangle := \int m_0(dx_0) \prod_{t'<t} (l_{t'}\mathcal{K}_{t'}) \\
&\langle \phi_{t+1}, m_{t+1} \rangle = \frac{\langle m_t, (l_t\mathcal{K}_t)\phi_{t+1} \rangle}{\langle m_t, l_t \rangle}
\end{aligned} \tag{11}$$

Thus, the new state is predicted by $x_{t+1} \sim m_{t+1}(\mathcal{X})$, where

$$m_{t+1} = \langle \left[\frac{l_t(x_t, \mathcal{L}_{x_t}; y_t)\mathcal{L}_{x_t}(dx_t)}{\int l_t(x'_t, \mathcal{L}_{x_t}; y_t)\mathcal{L}_{x_t}(dz'_t)} \right], \mathcal{K}_t \rangle \tag{12}$$

2) Decomposition of the State Space: We define the state space of the entire system as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where node $v \in \mathcal{V}$ represents observed zones, and edges $\mathcal{E}$ indicate their relationships. When the system cardinality $\mathcal{V}$ is huge, the signal becomes high-dimension. We encounter big data problem in two aspects: one is the computational burden, the other is error accumulation and propagation. Assume the

graph is sparse $\|\mathcal{E}\| \ll \|\mathcal{V}\|^2$, and the unconnected nodes are independent with each other; each signal-observation (x_t, y_t) has a projection on the nodes $(x_{vt}, y_{vt})_{v\in\mathcal{V}}$. The set of nodes $\mathcal{V}$ can be decomposed into o highly independent components:

$$\mathcal{V} := \mathcal{V}_1 \oplus \mathcal{V}_2 \oplus \ldots \oplus \mathcal{V}_o \tag{13}$$

Thus, the signal-observation space over the network can be decomposed as $\prod_{i=1}^{o} \mathcal{X}_{v\in\mathcal{V}_i}$ and $\prod_{i=1}^{o} \mathcal{Y}_{v\in\mathcal{V}_i}$. For node v, define its adjacent set as $\mathcal{N}(v)$. The state evolution at v is independent of other nodes except $\mathcal{N}(v)$. Since sampling, prediction and correction are performed within these highly independent zones $\mathcal{V}_i$ in parallel, we can reduce the computational cost and control the error propagation.

B. An Online Learning Problem

In the Sequential Monte Carlo approaches with importance sampling, higher weights are assigned to more confident estimations. However, in most cases we cannot directly observe estimation errors, since the observations are noisy, partial and unreliable. Here, we model it as an online supervised learning problem and derive a method to construct a confident evaluator for each candidate hypothesis. Suppose (x_t, y_t) is a pair of state and observation at time t, $x_t \in \mathbb{R}^m$, $y_t \in \mathbb{R}$, we learn a regression function $f : \mathbb{R}^m \to \mathbb{R}$ from a set of i.i.d. samples $(x_{i,t}, y_{i,t})$ drawn from an underlying joint distribution $\mathcal{P}$, $i \in \{1, \ldots, n\}$. For a new hypothesis x_j, the confident estimator corresponds to output of the regression function $f(x_j)$. We normalize the confident value to $(0, 1)$ so that it can be interpreted as the probability that x_j being an optimal estimation. Thus, sample weights can be updated by the regression function.

$\mathcal{P}$ can be assumed to be some parametric distributions (e.g. mixture of Gaussians), or to be distribution-free with some mild conditions such as smoothness and compactness. Then f is learnt by minimizing the regression error:

$$\min_f \sum_i^n L(f(x_i), y_i) + \lambda\Omega(f) \tag{14}$$

Here we use square error $L(f(x), y) = (f(x) - y)^2$, linear regression function $f(x) = w^T x$ and L^2-norm $\Omega(f) = \|w\|^2$. Thus, the learning problem becomes ridge regression:

$$\min_w \sum_i^n (w^T x_i - y_i)^2 + \lambda\|w\|^2 \tag{15}$$

it has closed-form solution [15]

$$w = (XX^H + \lambda I)^{-1} Xy \tag{16}$$

and the dual space solution is

$$\begin{aligned} w &= \sum_i^n \alpha_i x_i = X\alpha \\ \alpha &= (G + \lambda I)^{-1} y \end{aligned} \tag{17}$$

where $X \in \mathbb{R}^{m\times n}$ is data matrix with x_i in column i, $y \in \mathbb{R}^{n\times 1}$ with elements y_i, $X^H = (X^*)^T$ is the Hermitian transpose, and $G = X^H X$ is the Gram matrix.

1) Generating Virtual Samples by Cyclic Shift: In standard Sequential Monte Carlo methods, samples are generated from the estimated posterior distribution to approximate the filtering density. Typically, the performance can be increased by adding more samples, but this will lead to high computational cost. On the other hand, supervised learning also needs enough training samples to improve the regression accuracy and generalization ability. There is always some tradeoff between increasing the sample size and keeping a low computational cost. Since the optimal states are limited, there are finite positive examples, but the number of negative samples is infinite. The traditional way is to randomly choose only a few samples [16], or by hard-negative mining [17]. Samples near the optimal state are labeled $+1$ and those far away are labeled -1 [18].

Here, we perform dense sampling in the space domain and generate sufficient training examples by translation of the original samples. Based on the property of circulant matrix, for each real positive example $x_i \in \mathbb{R}^m$, all possible virtual negative samples are generated by cyclic shift and stored in the circulant data matrix $\mathcal{C}(x_i)$. Instead of binary labeling, we use continuous labels: set y to be 1 at the optimal state position, and gradually decay to 0 as distance increases. Based on the properties of circulant matrix (Equation 3), the optimal solution of ridge regression expressed in (15) becomes:

$$w = \mathcal{F}^{-1}\left(\frac{\mathbf{x} \circ \mathbf{y}}{\mathbf{x}^* \circ \mathbf{x} + \lambda}\right) \tag{18}$$

The dual solution is

$$\alpha = \mathcal{F}^{-1}\left(\frac{\mathbf{y}}{\mathbf{g}^{xx} + \lambda}\right) \tag{19}$$

where $\mathbf{x}, \mathbf{y}$ are fourier transform of x, y, and $\mathbf{g}^{xx}$ is the first row of the Gram matrix $G = X^H X = \mathcal{C}(g^{xx})$. Since the computation contains only FFT and element-wise production, the cost is reduced from $\mathcal{O}(m^3)$ to $\mathcal{O}(m \log m)$.

2) Batch Correction in Fourier Domain: In the correction stage, we want to assign larger weights to more confident hypotheses, and use the weighted sum as new estimation. For dense sampling, the naive way is to evaluate the hypotheses iteratively. In statistics, correlation indicates dependence or association of two sets of data. Let x and x' be two vectors, the correlation is defined as a kernel vector $k^{xx'}$ with elements

$$k_j^{xx'} = \kappa(T^{j-1}x, x') \tag{20}$$

where κ could be RBF kernel or Gaussian kernel. Instead of calculating single estimation z_i iteratively, we perform batch correction on the entire set of candidate hypotheses z. We update the sample weights by computing the regression output

$$weight \propto f(z) = \mathcal{F}^{-1}(\mathcal{F}(k^{xz}) \circ \mathcal{F}(\alpha)) \tag{21}$$

C. Correlative Mean-Field Filtering Framework

There are four operations in the filtering framework: decomposition D, sampling S^n, prediction P and correction C, where n is the sample size. As shown in Figure 3, the correlative mean-field filter (CMF) algorithm is performed iteratively to generate the estimated filtering distribution $\hat{m}(\mathcal{X})$

$$\hat{m}_{t+1}^n(\mathcal{X}) = C_{t+1}DPS^n,\ \hat{m}_t^n(\mathcal{X}) =: \hat{O}_{t+1}^n \hat{m}_t^n(\mathcal{X}) \tag{22}$$

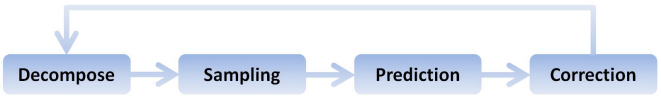


Fig. 3: Flow diagram of the CMF algorithm

IV. Experiments

A. Illustration of the Mean-Field filters

Problem Formulation: Consider the following state dynamics of mean-field type where the mean-field term is through the second moment:

$$\begin{aligned} &x_{k+1} = x_k + \alpha x_k(1 - E[x_k^2])\Delta t + \sigma x_k\sqrt{k\Delta t}N(0,1) \\ &x_0 \text{ given in } L^2, \\ &y_k = x_k^2 + \beta x_k + \sqrt{k\Delta t}N(0,1) \end{aligned} \tag{23}$$

where $N(0,1)$ denotes a normalized centered Gaussian distribution and Δt is the time step, α, β are positive real numbers.

Proposition 1. *Assume that the initial random variable x_0 satisfies $\mathbb{E}[x_0^2] < +\infty$. Then, the second moment can be computed recursively (online)*

$$\begin{aligned} E[x_{k+1}^2] &= \{(1 + \alpha\Delta t(1 - Ex_k^2))^2 + \sigma^2 k\Delta t\}E[x_k^2] \\ &= E[x_0^2]\prod_{l=1}^{k}\{(1 + \alpha\Delta t(1 - Ex_l^2))^2 + \sigma^2 l\Delta t\} \end{aligned} \tag{24}$$

The second moment of x_k is bounded by

$$E[x_k^2] \le E[x_0^2]c_{k\Delta t}, \tag{25}$$

for some $0 < c_{k\Delta t} < +\infty$, for any k such that $k\Delta t \le T$. Moreover the observation process y_k solves

$$Ey_k = Ex_k^2 + \beta Ex_k.$$

Proof. The proof follows directly from state dynamics model. □

Steady State Analysis: The deterministic system is obtained when σ vanishes. It is given by

$$\begin{aligned} &x_{k+1} = x_k + \alpha\Delta t x_k(1 - x_k^2) = (1 + \alpha\Delta t)x_k - \alpha\Delta t x_k^2 \\ &x_0 \text{ given in } \mathbb{R} \end{aligned} \tag{26}$$

Since $\alpha > 0,\ \Delta t > 0$, the set of steady states yields the zeros of $x(1 - x^2)$. It has one unstable equilibrium at $x = 0$ and two stable equilibria ($x = \pm 1$). With parameters $\Delta t = 0.1$, $\alpha = 1$, $\sigma = 10^{-3}$, $\beta = 10^{-2}$, $\eta = 10^{-2}$, the evolution of the system in total time $T = 30$ is shown in Figure 4a.

1) Mean-Field Free Model: We use particle filter (PF) as a mean-field free model to estimate the system state. Here we substitute the expectation term $E[x_t^2]$ directly with x_t^2, thus the system dynamic becomes mean-field free:

$$dx = \alpha x(1 - x^2)dt + \sigma x dw_t \tag{27}$$

In discrete form it is:

$$x_{k+1} = x_k + \alpha x_k(1 - x_k^2)\Delta t + \sigma x_k\sqrt{k\Delta t}N(0,1) \tag{28}$$

For each particle, it evolves independently, and the optimal states are estimated by the weighted average based on the observations. However, when β is small, either the quadratic term or the noise will be dominant. Therefore the observation is misleading and the sign information of x_k is missing, which causes the failure of PF estimator (Figure 5a). Take expectation on both sides of Equation (28) we have:

$$\begin{aligned} E[x_{k+1}^2] &= E[x_k + \alpha x_k(1 - x_k^2)\Delta t + \sigma x_k\sqrt{k\Delta t}N(0,1)]^2 \\ &= E[x_k + \alpha x_k\Delta t - \alpha x_k^3\Delta t + \sigma x_k\sqrt{k\Delta t}N(0,1)]^2 \\ &= E[(x_k + \alpha x_k\Delta t - \alpha x_k^3\Delta t)^2] + E[(\sigma x_k\sqrt{k\Delta t})^2] \\ &= (1 + \alpha\Delta t)^2 E[x_k^2] + \alpha^2\Delta t^2 E[x_k^6] + k\Delta t\sigma^2 E[x_k^2] \\ &\quad - 2\alpha\Delta t(1 + \alpha\Delta t)E[x_k^4] \end{aligned} \tag{29}$$

When the system is mean-field dependent, PF is risky because $E[x_{k+1}^2]$ evolves in terms of the unstable factor $E[x_k^6]$ and $E[x_k^4]$.

2) Mean-Field Dependent Model: In mean-field dependent model, we need to compute the expectation term $E[x_k^2]$ explicitly:

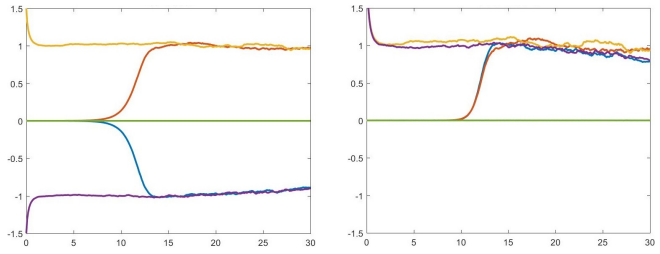
$$\begin{aligned} E[x_{k+1}^2] &= E[(x_k + \alpha x_k(1 - E[x_k^2])\Delta t \\ &\quad + \sigma x_k\sqrt{k\Delta t}N(0,1))^2] \\ &= E[((1 + \alpha\Delta t)x_k - \alpha\Delta t x_k E[x_k^2] \\ &\quad + \sigma\sqrt{k\Delta t}N(0,1)x_k)^2] \\ &= E[(1 + \alpha\Delta t)^2 x_k^2 - 2\alpha\Delta t x_k^2(1 + \alpha\Delta t)E[x_k^2] \\ &\quad + \alpha^2\Delta t^2 x_k^2(E[x_k^2])^2]^2 + E[\sigma^2 k\Delta t N^2(0,1)x_k^2]^2 \\ &= [(1 + \alpha\Delta t)^2 + \sigma^2 k\Delta t]E[x_k^2] + \alpha^2\Delta t^2 E^3[x_k^2] \\ &\quad - 2\alpha\Delta t(1 + \alpha\Delta t)E^2[x_k^2] \end{aligned} \tag{30}$$

Thus, $E[x_k^2]$ can be generated offline, and then used for updating x_{k+1}. Using mean-field filter (MF), the estimator will not diverge (Figure 5a).

However, sometimes we cannot compute the mean-field term directly, the second approach is to estimate the expectation term by generating virtual particles. The expected value $E[x_k^2]$ will be approximated by the ensemble mean $\frac{1}{M}\sum_{j=1}^{M} x_{j,k}^2$, where M is the number of virtual particles. The virtual particle system is given by:

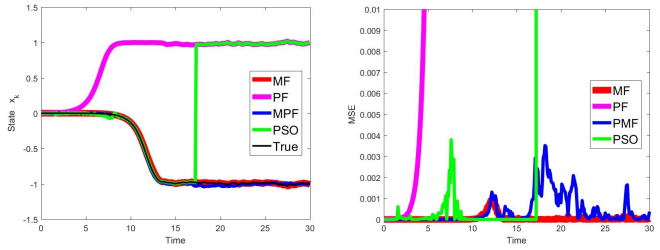
$$\begin{aligned} x_{i,k+1} &= x_{i,k} + \alpha x_{i,k}(1 - E[x_{i,k}^2])\Delta t + \sigma x_{i,k}\sqrt{k\Delta t}N(0,1) \\ &= x_{i,k} + \alpha x_{i,k}(1 - \frac{1}{M}\sum_{j=1}^{M} x_{j,k}^2)\Delta t \\ &\quad + \sigma x_{i,k}\sqrt{k\Delta t}N(0,1) \\ &= x_{i,k} + \alpha x_{i,k}(1 - \frac{1}{M}x_{i,k}^2 - \frac{1}{M}\sum_{j\neq i} x_{j,k}^2)\Delta t \\ &\quad + \sigma x_{i,k}\sqrt{k\Delta t}N(0,1) \end{aligned} \tag{31}$$

where $x_{i,k}$ denotes the outer real particles, and $x_{j,k}$ denotes the inner virtual particles, $i, j \in \{1, \ldots, M\}$. At each iteration, the virtual particles have the same distribution as $x_{i,k}$. The particle approximated mean-field filter (PMF) is stable, but



(a) System State (b) Observation

Fig. 4: The evolution of system state with five different initial states: $x_0 = 0, \pm 1.5, \pm 10^{-5}$



(a) State Estimator (b) Mean Squared Error

Fig. 5: Time evolution of the real state and their estimators using MF, PF, PMF and PSO [9]

cannot be implemented in parallel as PF, since at each iteration communication is necessary for virtual particle generation. We can see in Figure 5b, the error of PMF is larger than MF, which is caused by ensemble approximation. Finally, we repeated the experiment 1000 times, and set particle number $M = 100$. The numerical results are summarized in TABLE I.

TABLE I: Numerical Results

Filters	Time (s)	Global MSE	Failure (%)	Model
PF	0.0093	29.82	50.6	mf-free
PSO	8.0392	1.178	50.4	mf-free
MF	0.0042	**0.0248**	**0**	mf-dependent
PMF	**0.0030**	0.6164	**0**	mf-dependent

B. Correlative Mean-Field filter in Vehicle Tracking

In this section we implement CMF on a real vehicle tracking application. The input data is video sequences captured by 12 traffic cameras mounted on a highway network in Maryland. They form a local traffic surveillance system where each camera monitoring a part of the road. Our goal is to track the positions of multiple vehicles simultaneously.

1) System Model: In video sequence, following a target given its initial location is a filtering problem [20]. In our case, the initial position is unknown. Since the camera is stationary, foreground detection [21] is performed as a pre-processing step to estimate the initial state distribution $m_0(\mathcal{X})$. System state $x_t = (p_t, s_t, v_t)^T$ denotes the vehicle position, size and velocity in frame t, and observation y_t is the region of interest. The state transition kernel $\mathcal{K}_t(x_t, \mathcal{L}_{x_t}; x_{t+1})$ is modeled by the motion law, thus the system dynamic is

$$\begin{aligned} p_{t+1} &= p_t + v_t + n_p \\ s_{t+1} &= s_t(E_s(p_{t+1})/E_s(p_t) + n_s) \\ v_{t+1} &= \alpha v_t + (1-\alpha)E_v(p_{t+1}) + n_v \end{aligned} \tag{32}$$

where n_p, n_s, n_v are random Gaussian noise. E_s, E_v are mean-field terms that represent the expected vehicle size s and velocity v at position p. The observation model $p(y_t|x_t, \mathcal{L}_{x_t})$ indicates the probability of an observed region being a target under tracking. We use correlative mean-field filter to estimate next frame state x_{t+1} based on current observation and vehicle appearance model obtained from an online learning process.

2) Tracking Pipeline: In the t^{th} frame, we first generate n samples from the estimated state distribution $\hat{m}_t(\mathcal{X})$. The i^{th} sample is represented by a data vector $x_{i,t} \in \mathbb{R}^m$, which can be pure pixel intensities or HOG features [22]. Its corresponding label $y_{i,t} \in \mathbb{R}$ is set to 1 at the estimated target position $\hat{p}_{0,t}$, and smoothly decay to 0 by a Gaussian function

$$y_{i,t} = \exp(-\frac{1}{\sigma^2}\|\hat{p}_{0,t} - p_{i,t}\|^2), \forall i = 1, \ldots, n \tag{33}$$

Then we train the vehicle appearance model by solving a ridge regression problem defined in (15). In frame $t+1$, new candidate positions are predicted by the transition dynamic. Then, each candidate is evaluated by comparing the region of interest with the vehicle appearance model. In correction stage, sample weights are set by the output of the regression function. Finally, new state distribution $\hat{m}_{t+1}(\mathcal{X})$ is updated by the histogram of the sample weights, and new target position $\hat{p}_{0,t+1}$ is estimated as the one with the maximum weight.

3) Mean-Field Terms: The mean-field terms E_s, E_v are initialized by taking expected value of the prior distribution, which is obtained from history data or human experience. During tracking, the mean-field terms are updated in the system dynamic with new estimations $\hat{s}_t, \hat{v}_t$.

In tracking, there are three explanations for the introduce of mean-field terms. First, in mean-field free models the system state evolves only based on the previous state, which may accumulate error in case the dynamic model is inaccurate. Second, there's only a small part of the image that contains vehicles, i.e. the road zone, while most false alarms are waving trees or moving clouds. Third, the vehicle's speed varies with lanes, e.g. the traffic flow is dense and slow near intersections. So the distribution of vehicle position and speed should reflect these intuitions and be spatially distinct. In light of these reasons, it is reasonable to learn state distributions in a data driven way and embed mean-field terms into system model.

4) Performance Evaluation: The performance is evaluated by average successful tracking rate [23]. We compare our approach with particle filter [24] modified by [19], where the particle number is $n = 1000$ (in CMF $n = 10$). TABLE II shows performance comparisons on 12 traffic videos. The average successful tracking rate of CMF reaches 86.3%, which is much higher than 81.5% of MSPF [19]. Experimental results in Figure 6 also demonstrate the advantage of our approach.

Fig. 6: Performance comparison; top row: MSPF [19], bottom row: CMF

TABLE II: Performance comparison on 12 video sequences.

Video	MSPF (%)	CMF (%)	Video	MSPF (%)	CMF (%)
#1	86.8	88.7	#7	71.8	84.7
#2	84.6	86.8	#8	87.6	89.0
#3	90.3	92.0	#9	83.5	85.5
#4	80.6	86.8	#10	80.4	83.0
#5	75.7	88.2	#11	73.9	78.5
#6	83.9	90.0	#12	79.4	82.4

V. Conclusion

This paper proposed a novel correlative mean-field filtering framework for sequential and spatial data processing. The framework has four stages: decomposition, sampling, prediction and correction. By incorporating the mean-field term into system transition model, a robust estimator is developed, which is verified in the simulation. We study the properties of circulant structure in space domain and build a bridge between filtering and supervised learning. Dense sampling and batch correction are performed in Fourier space, which greatly increases the sample size with almost the same computation cost. Experimental results on real video data show that the proposed algorithm performs favorably against the state-of-the-art method.

References

[1] M. Dashti and A. M. Stuart, "The Bayesian Approach To Inverse Problems," *ArXiv e-prints*, Feb. 2013.

[2] A. Bain and D. Crisan, "Fundamentals of stochastic filtering," 2009.

[3] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Basic Engineering*, vol. 82, no. 1, pp. 35–45, Mar 1960.

[4] E. Alpaydin, *Introduction to Machine Learning*, MIT Press, 2010.

[5] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," *IEEE Trans. Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002.

[6] S. Pequito, A. P. Aguiar, B. Sinopoli, and D. A. Gomes, "Nonlinear estimation using mean field games," in *International Conference on NETwork Games, Control and Optimization*, Oct 2011, pp. 1–5.

[7] H. Tembine, R. Tempone, and P. Vilanova, "Mean-field learning: a survey," *CoRR*, vol. abs/1210.4657, 2012.

[8] G. Evensen, *Data Assimilation: The Ensemble Kalman Filter*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[9] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Neural Networks, 1995. Proceedings., IEEE International Conference on*, vol. 4, Nov 1995, pp. 1942–1948 vol.4.

[10] J. Gao and H. Tembine, "Distributed mean-field-type filters for big data assimilation," in *IEEE Conf. on Data Science and Systems*, Sydney, Australia, 2016, pp. 1446–1453.

[11] J. Gao and H. Tembine, "Distributed mean-field-type filter for vehicle tracking," in *American Control Conference (ACC)*, Seattle, USA, 2017.

[12] I. Kra and S. R. Simanca, "On circulant matrices," *Notices Amer. Math. Soc.*, vol. 59, no. 3, pp. 368–377, 2012.

[13] J. F. Henriques, "Circulant structures in computer vision," Ph.D. dissertation, Univ. of Coimbra, Coimbra, Portugal, 2016.

[14] R. M. Gray, "Toeplitz and circulant matrices: A review," *Foundations and Trends in Communications and Information Theory*, vol. 2, no. 3, pp. 155–239, 2006.

[15] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge University Press, 2004.

[16] B. Babenko, M. H. Yang, and S. Belongie, "Robust object tracking with online multiple instance learning," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1619–1632, Aug 2011.

[17] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, Sept 2010.

[18] B. Schlkopf and A. Smola, "Learning with kernels: Support vector machines, regularization, optimization, and beyond," *IEEE Transactions on Neural Networks*, vol. 16, no. 3, pp. 781–781, May 2005.

[19] G. Shabat, Y. Shmueli, A. Bermanis, and A. Averbuch, "Accelerating particle filter using randomized multiscale and fast multipole type methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 7, pp. 1396–1407, July 2015.

[20] H. Yang, L. Shao, F. Zheng, L. Wang, and Z. Song, "Recent advances and trends in visual tracking: A review," *Neurocomputing*, vol. 74, no. 18, pp. 3823 – 3831, 2011.

[21] K. Wang, Y. Liu, C. Gou, and F. Y. Wang, "A multi-view learning approach to foreground detection for traffic surveillance applications," *IEEE Trans. Vehicular Technology*, vol. 65, no. 6, pp. 4144–4158, 2016.

[22] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, June 2005, pp. 886–893 vol. 1.

[23] A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah, "Visual tracking: An experimental survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 7, pp. 1442–1468, July 2014.

[24] C. Hue, J. P. L. Cadre, and P. Perez, "Tracking multiple objects with particle filtering," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 38, no. 3, pp. 791–812, Jul 2002.