Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence

Monte Carlo Filtering Using Kernel Embedding of Distributions

Motonobu Kanagawa[†‡] and **Yu Nishiyama**[‡] and **Arthur Gretton**[§] and **Kenji Fukumizu**[†‡]
†Department of Statistical Science, Graduate University for Advanced Studies, Tokyo, 190-8562, Japan
‡The Institute of Statistical Mathematics, Tokyo, 190-8562, Japan
{*kanagawa, nishiyam, fukumizu*}*@ism.ac.jp*
§Gatsby Computational Neuroscience Unit, University College London, WC1N 3AR, United Kingdom
arthur.gretton@gmail.com

Abstract

Recent advances of kernel methods have yielded a framework for representing probabilities using a reproducing kernel Hilbert space, called kernel embedding of distributions. In this paper, we propose a Monte Carlo filtering algorithm based on kernel embeddings. The proposed method is applied to state-space models where sampling from the transition model is possible, while the observation model is to be learned from training samples without assuming a parametric model. As a theoretical basis of the proposed method, we prove consistency of the Monte Carlo method combined with kernel embeddings. Experimental results on synthetic models and real vision-based robot localization confirm the effectiveness of the proposed approach.

1 Introduction

Kernel methods traditionally have been used for constructing nonlinear learning machines from linear learning approaches (Schölkopf and Smola 2002). Recent advances of kernel methods have yielded a framework for nonparametric statistical inference called *kernel embedding of distributions* (Smola et al. 2007; Sriperumbudur et al. 2010; Song, Fukumizu, and Gretton 2013). This approach represents probability distributions by embedding into a *reproducing kernel Hilbert space (RKHS)*, on which we conduct statistical inference. Kernel embedding has been successfully applied to a wide variety of applications such as statistical testing (Gretton et al. 2012; 2008), time series analysis (Song et al. 2009; 2010; Fukumizu, Song, and Gretton 2011), belief propagation (Song, Gretton, and Guestrin 2010; Song et al. 2011), and reinforcement learning (Grünewälder et al. 2012b; Nishiyama et al. 2012). By virtue of kernel methods, this approach enables us to design nonparametric inference methods effective for high-dimensional and structured data with strong nonlinear dependence structures.

This paper proposes a Monte Carlo filtering algorithm for state-space models based on kernel embeddings, which we call *Kernel Monte Carlo filter*. Our method generates samples from the transition model, as in particle filters (Doucet, Freitas, and Gordon 2001). Our contribution to the kernel method is that we introduce a Monte Carlo method based on kernel embeddings. Importantly, this enables us to combine parametric models (e.g. transition models of state-space models) with nonparametric models learned with kernel embeddings (e.g. observation models). As a basis for the proposed algorithm, we provide a theoretical analysis for the sampling method combined with kernel embeddings.

Our filtering method is aimed at the setting where (1) sampling from the transition model $p(x_t|x_{t-1})$ is possible, while (2) the observation model $p(z_t|x_t)$ is unknown, even in a parametric form (Figure 1). Here, x_t and z_t denote the state and observation at time t, respectively. We assume that training samples $(X_1, Z_1), \dots (X_n, Z_n)$ for the observation model are available in a training phase. Our method learns the unknown observation model nonparametrically from the data using Kernel Bayes' rule (Fukumizu, Song, and Gretton 2011; 2014), which is a general kernel embedding approach for Bayesian inference. The proposed method can be applied to any domains for which kernels are defined, such as images, texts, graphs, etc. Moreover, by virtue of kernel embeddings, it is effective for problems that involve strong nonlinear dependency structures between the state and observation.

Applications of the proposed method can be found in robotics. For example, consider localization of a mobile robot from its vision images (Dellaert et al. 1999; Se, Lowe, and Little 2001). In this task, the state x_t corresponds to the robot's position, and the observation z_t its vision image. The localization problem is then reduced to a filtering problem, namely estimation of the posterior distribution over the position $p(x_t|z_{1:t})$ given a sequence of images $z_{1:t} := (z_1, \dots, z_t)$ for each time t. The transition model $p(x_t|x_{t-1})$ in this case corresponds to the robot's motion model. On the other hand, the observation model $p(z_t|x_t)$ is a conditional probability on the vision image given the robot's position. Since the vision strongly depends on the environment around the robot such as the structure of the rooms and building, it is basically challenging to define an appropriate parametric model for the observation model. Thus it can be considered unknown. However, we can obtain training samples of position-vision pairs $\{(X_i, Z_i)\}_{i=1}^n$ by using more expensive sensors before the test run (Quigley et al. 2010). The same situation can be found in signal-strength-based location estimation problems (Haeberlen et al. 2004; Ladd et al. 2002; Bahl and Padmanabhan 2000), for exam-

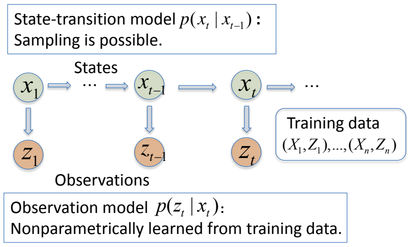

State-transition model $p(x_t | x_{t-1})$:
Sampling is possible.
States
x_1
x_{t-1}
x_t
z_1
z_{t-1}
z_t
Training data
$(X_1,Z_1),...,(X_n,Z_n)$
Observations
Observation model $p(z_t | x_t)$:
Nonparametrically learned from training data.

Figure 1: The setup of the paper.

ple.

This paper is organized as follows. We first review related works in Section 2. We review the kernel embedding approach in Section 3, and then propose the kernel Monte Carlo filter in Section 4. We theoretically analyze the Monte Carlo method in Section 5. We show experimental results on synthetic data and a real robot localization task in Section 6.

2 Related Work

As stated, this paper deals with the setup such that (1) sampling from the transition model $p(x_t|x_{t-1})$ is possible, while (2) the observation model $p(z_t|x_t)$ is to be learned from training samples $\{(X_i, Z_i)\}_{i=1}^n$ nonparametrically. Note that standard filtering methods such as particle filters (Doucet, Freitas, and Gordon 2001) cannot be applied to this situation directly, since they assume that the density of the observation model can be computed.

There are methods based on particle filters for dealing with this setup. These methods learn the unknown observation model with the k-nearest neighbors approach (Vlassis, Terwijn, and Kröse 2002) or Gaussian process regression (Ferris, Hähnel, and Fox 2006) from the training samples. Then, they combine the learned observation model with the transition model to perform particle filtering.

There also exists a related but different setting such that the transition model is also to be learned from transition examples of the state. For this setting, nonparametric filters using kernel embeddings (Song et al. 2009; Fukumizu, Song, and Gretton 2011) or Gaussian processes (Ko and Fox 2009; Deisenroth, Huber, and Hanebeck 2009) were proposed.

Note that while there are particle filters designed for "intractable" observation models (Rossi and Vila 2009; Jasra et al. 2012), the setting considered in these works are different from the one in this paper: they assume that the observation model is known and sampling is possible while its density function is not available.

3 Kernel Embedding of Distributions

Here, we briefly review the kernel embedding approach. For general introduction to the field, we refer to the tutorial papers (Smola et al. 2007; Song, Fukumizu, and Gretton 2013).

Let $\mathcal{X}$ be a space and $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a positive definite kernel such that $\sum_{i=1}^n \sum_{j=1}^n c_i c_j k(X_i, X_j) \geq 0$ holds for any $n \in \mathbb{N}, c_1, \ldots, c_n \in \mathbb{R}$ and $X_1, \ldots, X_n \in \mathcal{X}$. Examples of positive definite kernels include the Gaussian kernel $k(x, x') = \exp(-\|x - x'\|^2/2\sigma^2)$ and the Laplacian kernel $k(x, x') = \exp(-|x - x'|/\sigma)$ on $\mathcal{X} = \mathbb{R}^d$, where $\sigma > 0$.

For any positive definite kernel k, there exists a Hilbert space $\mathcal{H}$ uniquely associated with the kernel, called the *reproducing kernel Hilbert space (RKHS)*, which consists of functions on $\mathcal{X}$ (Schölkopf and Smola 2002). It is known that the reproducing property holds for $\mathcal{H}$: for any $x \in \mathcal{X}$ and $f \in \mathcal{H}$, we have $\langle k(\cdot, x), f \rangle_{\mathcal{H}} = f(x)$, where $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ denotes the inner-product of $\mathcal{H}$. Here, $k(\cdot, x) =: \phi(x)$ is the (possibly infinite-dimensional) feature vector of x. Then we can compute the inner product between feature vectors by $\langle \phi(x), \phi(x') \rangle_{\mathcal{H}} = k(x, x')$ using the reproducing property.

Let $\mathcal{P}$ be the set of all probability distributions on $\mathcal{X}$. Then we represent any probability distribution $P \in \mathcal{P}$ as an embedding into $\mathcal{H}$ defined by (Smola et al. 2007)

$$m_P := \mathbf{E}_{X \sim P}[\phi(X)] = \int \phi(x) dP(x) \in \mathcal{H}. \qquad (1)$$

Namely, we represent the distribution P by the expectation of the feature vector $\phi(x)$. We refer to $m_P \in \mathcal{H}$ as the *kernel embedding* of distribution P. We have a reproducing property for m_P in terms of expectation: for any $f \in \mathcal{H}$, we have $\langle f, m_P \rangle_{\mathcal{H}} = \mathbf{E}_{X \sim P}[f(X)]$ (Smola et al. 2007).

If the map $\mathcal{P} \to \mathcal{H} : P \to m_P$ is injective, i.e. $m_P = m_Q$ implies $P = Q$, then the kernel k used for embedding is called *characteristic* (Fukumizu et al. 2008). In other words, if we use a characteristic kernel, m_P uniquely identifies the distribution P. Thus, if our objective is to estimate distribution P from data, it suffices to estimate the corresponding embedding m_P. Examples of characteristic kernels include the Gaussian and Laplacian kernels. Other examples of characteristic kernels can be found in (Sriperumbudur et al. 2010). In the following, we assume that kernels are characteristic.

Let $\{X_1, \ldots, X_n\} \subset \mathcal{X}$ be data points. In general, the embedding m_P is estimated in the form of a weighted sum of feature vectors $\phi(X_i)$ of the data

$$\hat{m}_P = \sum_{i=1}^n w_i \phi(X_i), \qquad (2)$$

where the (possibly negative) weights $w_i \in \mathbb{R}$ are computed by an estimator of m_P (Song, Fukumizu, and Gretton 2013). For example, if X_i are i.i.d drawn from P, then $w_i = 1/n$ gives a consistent estimator with the rate $\|m_P - \hat{m}_P\|_{\mathcal{H}} = O_p(n^{-1/2})$, where $\|\cdot\|_{\mathcal{H}}$ denotes the norm of the RKHS $\mathcal{H}_{\mathcal{X}}$ (Smola et al. 2007).

On the other hand, the data $\{X_1, \ldots, X_n\}$ may come from joint samples $\{(X_1, Z_1), \ldots, (X_n, Z_n)\}$ generated from some joint distribution, where Z_i belongs to another space $\mathcal{Z}$. Then, the objective may be to estimate the embedding of a conditional probability $p(x|z)$, for example. In such cases, algorithms such as the conditional kernel embedding (Song et al. 2009; Grünewälder et al. 2012a) and Kernel Bayes' rule (Fukumizu, Song, and Gretton 2011; 2014), which we will use in the proposed method, can be used for computing the weights w_i in (2). These algorithms compute the weights with simple linear algebraic operations.

4 Kernel Monte Carlo Filter

First, we define the notation and review the setup (also see Figure 1 and the third and fourth paragraphs in Section 1).

Let $\mathcal{X}$ and $\mathcal{Z}$ be the spaces of state and observation, respectively. Let $x_t \in \mathcal{X}$ and $z_t \in \mathcal{Z}$ be the state and observation at time t, respectively.

Let $p(x_t|x_{t-1}, u_t)$ be the transition model, where u_t denotes the control at time t. Here, we include u_t to explicitly describe the sampling procedure. We assume that we can generate samples from the transition model. Let $p(z_t|x_t)$ be the *unknown* observation model.

We assume that training samples $\{(X_i, Z_i)\}_{i=1}^n \subset \mathcal{X} \times \mathcal{Z}$ for the observation model are available. Let $z_{1:t} := (z_1, \ldots, z_t)$ be the sequence of test observations. Then the task of filtering is to estimate the posterior distribution over the state $p(x_t|z_{1:t})$ for each time $t = 1, \ldots, T$, where $T \in \mathbb{N}$, by exploiting the training samples and the transition model. Note that we cannot observe the ground-truth state x_t.

Let $k_\mathcal{X} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a kernel on $\mathcal{X}$, $\phi(x) := k_\mathcal{X}(\cdot, x)$ be the feature vector of point $x \in \mathcal{X}$, and $\mathcal{H}_\mathcal{X}$ be the RKHS associated with $k_\mathcal{X}$. Let $k_\mathcal{Z} : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}$ be a kernel on $\mathcal{Z}$. Then, our filtering problem is to estimate the kernel embedding of the posterior distribution for each time t:

$$m_{x_t|z_{1:t}} := \int \phi(x_t) p(x_t|z_{1:t}) dx_t \in \mathcal{H}_\mathcal{X}.$$

We omit the controls u_t from the notation of the posterior distributions and the kernel embeddings for simplicity.

Proposed Algorithm

As in general filtering methods, the proposed method iterates *prediction* and *correction steps*. The procedure is summarized in Algorithm 1, where p_{init} denotes an initial distribution over the state. Figure 2 describes the proposed method at one time step.

Prediction Step Suppose that we have already estimated the embedding of the posterior distribution $m_{x_{t-1}|z_{1:t-1}} := \int \phi(x_{t-1}) p(x_{t-1}|z_{1:t-1}) dx_{t-1}$. Let

$$\hat{m}_{x_{t-1}|z_{1:t-1}} := \sum_{i=1}^n w_{t-1,i} \phi(X_i) \tag{3}$$

be the estimate, where $w_{t-1,i} \in \mathbb{R}$. Note that X_i are the points in the training data $\{(X_i, Z_i)\}_{i=1}^n$. This is because $\hat{m}_{x_{t-1}|z_{1:t-1}}$ is estimated using Kernel Bayes' rule in the correction step, as will be seen later.

Let $p(x_t|z_{1:t-1})$ be the prior distribution over the state at time t. In this step, we estimate its kernel embedding $m_{x_t|z_{1:t-1}} := \int \phi(x_t) p(x_t|z_{1:t-1}) dx_t$. Given a control u_t, first we draw samples from the transition model for each X_i

$$X_{t,i} \sim p(x_t|X_i, u_t), \quad i = 1, \ldots, n.$$

Then we estimate the embedding as the sum of feature vectors for the sampled points with the same weights as (3):

$$\hat{m}_{x_t|z_{1:t-1}} := \sum_{i=1}^n w_{t-1,i} \phi(X_{t,i}). \tag{4}$$

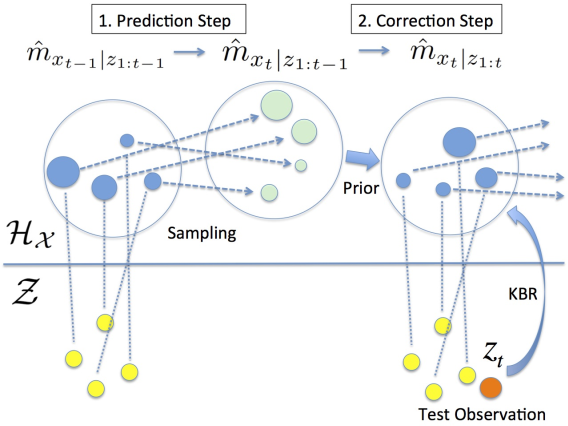

1. Prediction Step
2. Correction Step
$\hat{m}_{x_{t-1}|z_{1:t-1}} \longrightarrow \hat{m}_{x_t|z_{1:t-1}} \longrightarrow \hat{m}_{x_t|z_{1:t}}$
Prior
$\mathcal{H}_\mathcal{X}$
Sampling
$\mathcal{Z}$
KBR
z_t
Test Observation

Figure 2: Proposed method at one time step. The pairs of blue and yellow circles indicate training samples $\{(X_i, Z_i)\}_{i=1}^n$. **1. Prediction step**: we generate samples from each state in the training samples using the transition model. Combined with the propagated weights (indicated by the size of circles), the prior embedding is estimated. **2. Correction step**: we estimate the posterior embedding by applying Kernel Bayes' rule (KBR) to the new observation, training samples, and the prior embedding estimate. The estimate is again given as a weighted sample expression for the training samples.

Note that while this sampling procedure is similar to the one in particle methods (Doucet, Freitas, and Gordon 2001), there exists a crucial difference: we are estimating the *kernel embedding* of the prior distribution, which is an element in the RKHS $\mathcal{H}_\mathcal{X}$. Thus we need a new convergence analysis for the estimate, which we will provide in the next section.

Correction Step Let z_t be a new observation. In this step, we estimate the embedding of the posterior distribution $m_{x_t|z_{1:t}} := \int \phi(x_t) p(x_t|z_{1:t}) dx_t$. To this end, we can use the estimated prior embedding (4) and the training samples $\{(X_i, Z_i)\}_{i=1}^n$ for the unknown observation model.

We employ Kernel Bayes' rule (Fukumizu, Song, and Gretton 2011; 2014), which is a consistent estimator of posterior embeddings in general Bayesian inference and applicable to our setting. Let $G_X := (K_\mathcal{X}(X_i, X_j)) \in \mathbb{R}^{n \times n}$ and $G_Z := (k_\mathcal{Z}(Z_i, Z_j)) \in \mathbb{R}^{n \times n}$ be the kernel matrices computed with the training data $\{(X_i, Z_i)\}_{i=1}^n$. Compute

$$\begin{aligned} \mathbf{m}_{x_t|z_{1:t-1}} &:= (\langle \hat{m}_{x_t|z_{1:t-1}}, \phi(X_j) \rangle_{\mathcal{H}_\mathcal{X}})_{j=1}^n \\ &= \left(\sum_{i=1}^n w_{t-1,i} k_\mathcal{X}(X_j, X_{t-1,i}) \right)_{j=1}^n \in \mathbb{R}^n, \qquad (5) \\ \mathbf{k}_Z(z_t) &:= (k_\mathcal{Z}(z_t, Z_j))_{j=1}^n \in \mathbb{R}^n. \qquad (6) \end{aligned}$$

Then Kernel Bayes' rule estimates the posterior embedding $m_{x_t|z_{1:t}}$ by the following formulas:

$$\hat{m}_{x_t|z_{1:t}} := \sum_{i=1}^n w_{t,i} \phi(X_i), \tag{7}$$

$$w_t := \Lambda G_Z ((\Lambda G_Z)^2 + \delta_n I_n)^{-1} \Lambda \mathbf{k}_Z(z_t) \in \mathbb{R}^n, \tag{8}$$

$$\Lambda := \mathrm{diag}((G_X + n\varepsilon_n I_n)^{-1} \mathbf{m}_{x_t|z_{1:t-1}}) \in \mathbb{R}^{n \times n}(9)$$

where $\mathrm{diag}(v)$ denotes the diagonal matrix with diagonal entries $v \in \mathbb{R}^n$ and $\varepsilon_n, \delta_n > 0$ the regularization coefficients.

Roughly, Kernel Bayes' rule can be interpreted as two-step nonparametric regression: the first step (9) encodes the prior embedding estimate (4) as matrix Λ. Then the second step (7)(8) estimates a regression function from z to $H_{\mathcal{X}}$, which corresponds to the embedding of the posterior distribution $m_{x_t|z_{1:t}}$.

As shown in (7), for each time, the posterior embedding is represented using the training samples $X_1, \ldots, X_n$. Thus, samples will not be spread over time. Therefore we do not need a resampling step, as opposed to particle methods (Doucet, Freitas, and Gordon 2001).

Point Estimation of the State Note that the output of the proposed algorithm is an estimate of the kernel embedding $\hat{m}_{x_t|z_{1:t}} = \sum_{i=1}^n w_{t,i}\phi(X_i)$, which is an element in the RKHS $\mathcal{H}_{\mathcal{X}}$. Here, we explain how to give a point estimate of the state from the estimate.

The posterior mean $\int x_t p(x_t|z_{1:t})dx_t$ can be estimated by the empirical average[1] $\sum_{i=1}^n w_{t,i}X_i$. We can also use the heuristic to estimate the state by computing the pre-image $\arg\min_{x\in\mathcal{X}} \|\phi(x) - \hat{m}_{x_t|z_{1:t}}\|_{\mathcal{H}_{\mathcal{X}}}$ (Song et al. 2009). Note that if the posterior distribution $p(x_t|z_{1:t})$ is highly multimodal, these methods may not work. For such situations, we can employ another heuristic to use a point with the maximum weight $X_{i_{\max}}$, where $i_{\max} := \arg\max_i w_i$.

Hyperparameters There exist hyperparameters in the proposed method: parameters in kernels $k_{\mathcal{X}}, k_{\mathcal{Z}}$ (such as the bandwidth parameter in a Gaussian kernel) and regularization coefficients ε_n, δ_n of Kernel Bayes' rule. We can select these parameters, for example, by dividing the training data into two-sequences and then applying two-fold cross validation. We can use the root mean squared errors (RMSE) between estimated and ground-truth states as evaluation criteria for validation.

Time Complexity Dominant parts are the matrix inversions in the correction step (8)(9), each of which costs $O(n^3)$ if naively computed. However, we can reduce them by applying low rank approximation to the involved matrices using methods such as incomplete Cholesky decomposition (Fine and Scheinberg 2001). Then the cost is reduced to $O(nr^2)$, where $r \ll n$ is the approximation rank. Note that the inversion $(G_X + n\varepsilon_n I_n)^{-1}$ in (9) can be computed before the test run since it only involves the training data.

5 Theoretical Analysis

This section provides a convergence analysis for the prediction step. Note that the consistency of the correction step is guaranteed by the convergence theorem of Kernel Bayes' rule (Fukumizu, Song, and Gretton 2011; 2014).

We consider a general setting; let $\mathcal{X}$ and $\mathcal{Y}$ be measurable spaces, $p_X(x)$ be a probability distribution on $\mathcal{X}$, $p(y|x)$ be a conditional probability on $\mathcal{Y}$ given $x \in \mathcal{X}$, and $p_Y(y)$ be a distribution on $\mathcal{Y}$ defined by $p_Y(y) := \int p(y|x)p_X(x)dx$. In the filtering setting, $\mathcal{X}$ and $\mathcal{Y}$ correspond to the state-spaces at time $t-1$ and t, respectively. Distributions $p_X(x)$, $p(y|x)$, and $p_Y(y)$ correspond to $p(x_{t-1}|z_{1:t-1})$, $p(x_t|x_{t-1}, u_t)$, and $p(x_t|z_{1:t-1})$, respectively.

Algorithm 1 Kernel Monte Carlo Filter

```
Input: Training data $\{(X_i, Z_i)\}_{i=1}^n$, test observations $\{z_j\}_{j=1}^T$, control inputs $\{u_j\}_{j=1}^T$.
Set $w_{0,i} = 1/n,\ i = 1, \ldots, n$.
for $t = 1$ to $T$ do
    if $t = 1$ then
        Generate $X_{1,i} \sim p_{\mathrm{init}},\ i = 1, \ldots, n$.
    else
        Generate $X_{t,i} \sim p(\cdot|X_i, u_t),\ i = 1, \ldots, n$.
    end if
    Calculate $\mathbf{m}_{x_t|z_{1:t-1}}$ (Eq. (5))
    Observe $z_t$ and calculate $\mathbf{k}_Z(z_t)$ (Eq. (6))
    Calculate $w_t \in \mathbb{R}^n$ (Eqs. (8)(9)).
end for
Output: Estimates of the posterior embeddings $\hat{m}_{x_t|z_{1:t}} = \sum_{i=1}^n w_{t,i}\phi(X_i), t = 1, \ldots, T$.
```

Let $k_{\mathcal{X}}$ and $k_{\mathcal{Y}}$ be kernels on $\mathcal{X}$ and $\mathcal{Y}$, and $\mathcal{H}_{\mathcal{X}}$ and $\mathcal{H}_{\mathcal{Y}}$ be their associated RKHSs, respectively. Denote by $\phi(x) := k_{\mathcal{X}}(\cdot, x) \in \mathcal{H}_{\mathcal{X}}$ and $\psi(y) := k_{\mathcal{Y}}(\cdot, y) \in \mathcal{H}_{\mathcal{Y}}$ the feature vectors. Assume that we are given an estimate of the kernel embedding $m_X := \int \phi(x)p_X(x)dx \in \mathcal{H}_{\mathcal{X}}$ by

$$\hat{m}_X := \sum_{i=1}^n w_i \phi(X_i), \qquad (10)$$

where $w_i \in \mathbb{R}$ are weights and $X_i \in \mathcal{X}$ are data points. In the filtering setting, this corresponds to $\hat{m}_{x_{t-1}|z_{1:t-1}}$. Then we generate samples from the conditional distribution

$$Y_i \sim p(dy|X_i), \quad i = 1, \ldots, n,$$

and estimate $m_Y := \int \psi(y)dp_Y(y) \in \mathcal{H}_{\mathcal{Y}}$ by

$$\hat{m}_Y := \sum_{i=1}^n w_i \psi(Y_i). \qquad (11)$$

Theorem 1 below shows that (11) is a consistent estimator of m_Y. Note that m_Y corresponds to the embedding of the prior distribution $m_{x_t|z_{1:t-1}}$ in the filtering setting. Thus the theorem shows the consistency of the prediction step. The proof is given in the supplementary materials.

Theorem 1. *Let $k_{\mathcal{X}}$ and $k_{\mathcal{Y}}$ be bounded characteristic kernels. Assume that $\hat{m}_X$ (10) satisfies $\mathbf{E}[\|\hat{m}_X - m_X\|_{\mathcal{H}_{\mathcal{X}}}^2] = O(n^{-b})$ and $\mathbf{E}[w^T w] = O(n^{-c})$ for some $b, c > 0$ as $n \to \infty$. Assume that the function $\theta(x, \tilde{x}) := \mathbf{E}_{Y\sim p(dy|x)}\mathbf{E}_{\tilde{Y}\sim p(d\tilde{y}|\tilde{x})}[k_{\mathcal{Y}}(Y, \tilde{Y})]$ satisfies $\theta \in \mathcal{H}_{\mathcal{X}} \otimes \mathcal{H}_{\mathcal{X}}$, where $\mathcal{H}_{\mathcal{X}} \otimes \mathcal{H}_{\mathcal{X}}$ denotes the RKHS of the product kernel $k_{\mathcal{X}} \otimes k_{\mathcal{X}}$ on $\mathcal{X} \times \mathcal{X}$. Then for $\hat{m}_Y$ (11) we have*

$$\mathbf{E}[\|\hat{m}_Y - m_Y\|_{\mathcal{H}_{\mathcal{Y}}}^2] = O(n^{-\min(b,c)}) \quad (n \to \infty).$$

[1] This is theoretically justified if the projection function $f_k : \mathbb{R}^d \to \mathbb{R}, x \to x_k$, where $k = 1, \ldots, d$, satisfies $f_k \in \mathcal{H}_{\mathcal{X}}$ (Smola et al. 2007). Otherwise we can use it as a heuristic.

Here we explain the theorem. First, we assume that $\hat{m}_X = \sum_{i=1}^n w_i \phi(X_i)$ converges to m_X with the rate $O(n^{-b/2})$ in expectation, as sample size n goes to infinity. Another assumption on $\hat{m}_X$ is that the squared sum of the weights $w^T w = \sum_{i=1}^n w_i^2$ converges to zero with the rate $O(n^{-c})$. Theorem 1 states that under these assumptions, $\hat{m}_Y$ converges to m_Y with the rate $O(n^{-\min(b,c)/2})$. Note that assumption $\theta \in \mathcal{H}_\mathcal{X} \otimes \mathcal{H}_\mathcal{X}$ is a technical one to simplify the assertion and the involved proof, and can be weakened by using approximation arguments of statistical learning theory, e.g., (Eberts and Steinwart 2013).

For instance, if X_i are i.i.d. drawn from p_X and $\hat{m}_X$ is given as $\frac{1}{n}\sum_{i=1}^n \phi(X_i)$, we have $b = 1, c = 1$ (Smola et al. 2007). Therefore $\hat{m}_Y$ converges to m_Y with the rate $O(n^{-1/2})$ in this case. We can also show that if $\hat{m}_X$ is given by the conditional kernel embedding (Song et al. 2009; Grünewälder et al. 2012a), which is a basis for Kernel Bayes' rule, a general upper-bound is given as $b = 1/4, c = 3/4$ (see the supplementary materials). Therefore, in this case $\hat{m}_Y$ converges to m_Y with the rate $O(n^{-1/8})$ at worst.

Note that Theorem 1 can be applied not only to the filtering setting, but also to other kernel embedding algorithms that may incorporate the sampling procedure. For example, we can naturally extend the kernel POMDP algorithm (Nishiyama et al. 2012) to the one with the sampling procedure. Theorem 1 can also provide a theoretical basis for such an extension.

6 Experiments

We compare the proposed method with existing algorithms applicable to the setting of the paper. Comparisons are done with (1) the particle filter with k-nearest approach (PF-NN) (Vlassis, Terwijn, and Kröse 2002), (2) the particle filter with Gaussian process regression (PF-GP) (Ferris, Hähnel, and Fox 2006), and (3) Kernel Bayes' rule filter (KBRF) (Fukumizu, Song, and Gretton 2011).

As mentioned in Section 2, PF-NN and PF-GP learn the observation model from training samples using the k-NN approach and GP-regression, respectively. We use an open-source code for GP-regression[2], and therefore omit comparison in computational time with PF-GP. KBRF is based on kernel embeddings, and applies Kernel Bayes' rule in the correction step. This method is designed for settings where the transition model is also to be learned from state-transition examples, and uses the kernel sum rule (Song et al. 2009) in the prediction step. Thus, we use KBRF as a baseline for the proposed method. Note that KBRF was shown to outperform the nonlinear Kalman filters (Fukumizu, Song, and Gretton 2011), and thus we do not compare the proposed method with them.

We fix the number of particles in PF-NN and PF-GP to 5000, since we did not observe any improvement with a larger number of particles in a preliminary experiment. We additionally generate state-transition examples for KBRF and fix the size to 1000.

We evaluate the performance of each algorithm by the root mean squared errors (RMSE) between estimated and ground-truth states. We also use RMSE in cross-validation to select hyperparameters of each method.

Synthetic Data Experiments

To generate synthetic data, we used the following state-space models, which are defined on $\mathcal{X} = \mathcal{Z} = \mathbb{R}$. Here $\mathbb{N}(\mu, \sigma^2)$ denotes the normal distribution with mean μ and variance σ^2. Recall that x_t, z_t, and u_t denote the state, observation, and control at time t, respectively.

Synthetic Model 1.

$$\begin{aligned} &x_1 = v_1,\ \ v_1 \sim \mathbb{N}(0, 1/(1-0.9^2)). \\ &x_t = 0.9x_{t-1} + 0.5u_t + 0.5v_t,\ \ v_t \sim \mathbb{N}(0,1). \\ &z_t = x_t + w_t,\ \ w_t \sim \mathbb{N}(0,1). \end{aligned}$$

Synthetic Model 2.

$$\begin{aligned} &x_1 = v_1,\ \ v_1 \sim \mathbb{N}(0, 1/(1-0.9^2)). \\ &x_t = 0.9x_{t-1} + 0.5u_t + 0.5v_t,\ \ v_t \sim \mathbb{N}(0,1). \\ &z_t = 0.5\exp(x_t/2)w_t,\ \ w_t \sim \mathbb{N}(0,1). \end{aligned}$$

Synthetic Model 3.

$$\begin{aligned} &x_1 = v_1,\ \ v_1 \sim \text{uniform}([-3,3]), \\ &a_t = x_{t-1} + u_t + 0.3v_t,\ \ v_t \sim \mathbb{N}(0,1), \\ &\text{if } |a_t| \leq 3:\ x_t = a_t, \quad \text{else}:\ x_t = -3. \\ &b_t = x_t + 0.5w_t,\ \ w_t \sim \mathbb{N}(0,1), \\ &\text{if } |b_t| \leq 3:\ z_t = b_t, \quad \text{else}:\ z_t = b_t - 6b_t/|b_t|. \end{aligned}$$

The model 1 is a linear Gaussian model. The transition model of the model 2 is same as that of the model 1, but the observation model is highly nonlinear with multiplicative noise. The model 3 is also highly nonlinear in the transition and observation models: states and observations near the interval $[-3, 3]$ may abruptly move to distant points.

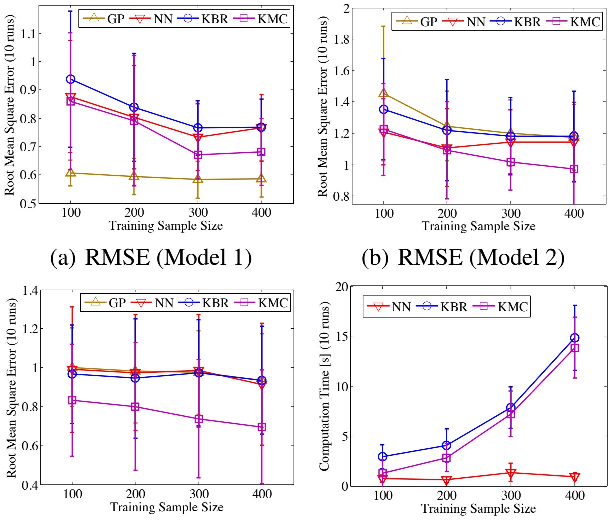
Controls u_t in each model were randomly generated from the normal distribution $u_t \overset{i.i.d.}{\sim} \mathbb{N}(0,1)$. We generated training samples $\{(X_i, Z_i)\}_{i=1}^n$ by sequentially running each model. Test observations $(z_1, \ldots, z_T)$ (and the corresponding ground truth states x_t, which cannot be observed by the filters) are also generated from the models. We set the length of the test sequence as $T = 100$.

The proposed method used Gaussian kernels for each of $\mathcal{X}$ and $\mathcal{Z}$. We also used Gaussian kernels for KBRF. We chose the hyperparameters in each filter by two-fold cross-validation by dividing the training data into two sequences. The hyperparameters in the GP-regressor of PF-GP are optimized by maximizing the marginal likelihood on the training data. We ran the experiment 10 times for each of different training sample size. Each method gives a point estimate of the ground-truth state by estimating the posterior mean.

The results are shown in Figure 3, in which GP, NN, KBR, and KMC correspond to PF-GP, PF-NN, KBRF, and the proposed method, respectively. For the model 1, PF-GP performed the best, since the model is additive Gaussian. Our method outperformed the competitors for the model 2 and 3, in which nonlinearity or non-Gaussianity of the observation

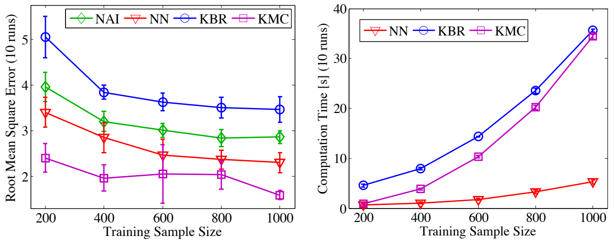
[2] http://www.gaussianprocess.org/gpml/code/matlab/doc/

models is higher than that of the model 1. This shows that our method is promising in the situations where the observation model cannot be easily modeled, which is the setting of the paper. Note that the large deviations in the results are due to the test data. We also conducted sign tests for the results, and the proposed method indeed significantly outperformed the competitors (see the supplementary materials). Computational time of the proposed method was competitive to that of KBRF, but slower than that of PF-NN due to the matrix inversion in the correction step. Results on computational time for the model 1 and 3 are omitted since they were almost the same as the model 2.



(a) RMSE (Model 1)
(b) RMSE (Model 2)
(c) RMSE (Model 3)
(d) Comp. Time [s] (Model 2)

Figure 3: Results of the synthetic experiments. GP, NN, KBR, and KMC in the figures correspond to PF-GP, PF-NN, KBRF, and the proposed method, respectively.



(a) RMSE
(b) Comp. Time

Figure 4: Results of robot localization. NAI, NN, KBR, and KMC in the figures correspond to the naive method, PF-NN, KBRF, and the proposed method, respectively.

Real Vision-Based Mobile Robot Localization

We conducted experiments on the vision-based mobile robot localization task. The goal is to sequentially estimate the position of a robot moving in a building, based on a sequence of vision images observed by the robot. Thus, the state-space $\mathcal{X}$ consists of the two-dimensional location and the orientation of the robot, i.e. $\mathcal{X} = \mathbb{R}^d \times [0, 2\pi]$. The observation space $\mathcal{Z}$ consists of vision images. In this experiment, we do not compare with PF-GP, since it assumes that the observations are real-valued vectors and therefore cannot be applied to this problem straightforwardly.

We used odometry motion model for the transition model $p(x_t|x_{t-1}, u_t)$, which is standard in robotics (Thrun, Burgard, and Fox 2005). In this model, u_t corresponds to the odmetry measurements. We used the algorithm in Table 5.6. of Thrun, Burgard, and Fox (2005), where we set the parameters to be a small value 0.1.

We used the spatial pyramid kernel (Lazebnik, Schmid, and Ponce 2006), which is a standard kernel for images, for $\mathcal{Z}$, with the parametrization that gives 4200 histograms for each image, as suggested by the authors. We used a Gaussian kernel[3] for $\mathcal{X}$. In this experiment, we also used the spatial pyramid kernel for PF-NN as a similarity measure of nearest neighbors search.

Datasets were taken from the COLD database (Pronobis and Caputo 2009), in which we used *Freiburg, Part A, Path 1, cloudy*. This dataset is made of three similar sequences, each of which consists of position-image pairs taken by a robot moving in a building. We used two of them for training, and the rest for test. The time resolution was set to 0.44 images per second.

The hyperparmeters of each method were chosen by two-fold cross-validation, using the two sequences of the training data. As a baseline for this task, we performed a naive method (NAI) that estimates the ground-truth state by the state in the training samples that has the closest observation to the test observation. Since posterior distributions are highly multimodal in this problem, we used the sample point with maximum weight for point estimation for the proposed method and KBRF. PF-NN estimated the state by the particle with maximum weight. We ran each experiment 10 times for each of different size of training data.

The results are shown in Figure 4. Our method (KMC) significantly outperformed the competitors in terms of RMSE. This shows that our method can effectively learn the complex observation model from the training data. Comparison with KBRF confirms the effectiveness of our sampling method combined with kernel embeddings.

7 Conclusions

We have presented a kernel embedding-based Monte Carlo filtering algorithm. The proposed method is aiming at the setting where sampling from the transition model is possible, while the observation model is unknown but its training samples are available. We proved the convergence of the sampling method with kernel embeddings. Applications of the proposed method can be found in robotics, such as localization problems.

Acknowledgements

We thank the anonymous reviewers for their helpful comments. This work was supported in part by JSPS Grant-in-Aid for Scientific Research on Innovative Areas 25120012.

[3] We projected the robot's orientation in $[0, 2\pi]$ onto the unit circle in $\mathbb{R}^2$. Then we defined a Gaussian kernel on $\mathbb{R}^4$.

References

Bahl, P., and Padmanabhan, V. N. 2000. RADAR: an in-building RF-based user location and tracking system. In *Proc. IEEE Infocom 2000*, volume 2, 775–784.

Deisenroth, M.; Huber, M.; and Hanebeck, U. 2009. Analytic moment-based Gaussian process filtering. In *Proc. 26th ICML.*

Dellaert, F.; Burgard, W.; Fox, D.; and Thrun, S. 1999. Using the condensation algorithm for robust, vision-based mobile robot localization. In *Proc. IEEE Intern. Conf. on Computer Vision and Pattern Recognition (CVPR).*

Doucet, A.; Freitas, N. D.; and Gordon, N. J., eds. 2001. *Sequential Monte Carlo Methods in Practice*. Springer.

Eberts, M., and Steinwart, I. 2013. Optimal regression rates for SVMs using Gaussian kernels. *Electron. J. Stat.* 7:1–42.

Ferris, B.; Hähnel, D.; and Fox, D. 2006. Gaussian processes for signal strength-based location estimation. In *Proc. Robotics: Science and Systems*.

Fine, S., and Scheinberg, K. 2001. Efficient SVM training using low-rank kernel representations. *J. Machine Learning Research* 2:243–264.

Fukumizu, K.; Gretton, A.; Sun, X.; and Schölkopf, B. 2008. Kernel measures of conditional dependence. In *Advances in NIPS 20*, 489–496.

Fukumizu, K.; Song, L.; and Gretton, A. 2011. Kernel Bayes' rule. In *Advances in NIPS 24*, 1737–1745.

Fukumizu, K.; Song, L.; and Gretton, A. 2014. Kernel Bayes' rule: Bayesian inference with positive definite kernels. *J. Machine Learning Research* 14:37533783.

Gretton, A.; Fukumizu, K.; Teo, C.; Song, L.; Schoelkopf, B.; and Smola, A. 2008. A kernel statistical test of independence. In *Advances in NIPS 20*, 585–592.

Gretton, A.; Borgwardt, K.; Rasch, M.; Schölkopf, B.; and Smola, A. 2012. A kernel two-sample test. *J. Machine Learning Research* 13:723–773.

Grünewälder, S.; Lever, G.; Baldassarre, L.; Patterson, S.; Gretton, A.; and Pontil, M. 2012a. Conditional mean embeddings as regressors. In *Proc. 29th ICML.*

Grünewälder, S.; Lever, G.; Baldassarre, L.; Pontil, M.; and Gretton, A. 2012b. Modeling transition dynamics in MDPs with RKHS embeddings. In *Proc. 29th ICML.*

Haeberlen, A.; Flannery, E.; Ladd, A. M.; Rudys, A.; Wallach, D. S.; and Kavraki, L. E. 2004. Practical robust localization over large-scale 802.11 wireless networks. In *Proc. 10th Intern. Conf. on Mobile computing and networking*, 70–84.

Jasra, A.; Singh, S. S.; Martin, J. S.; and McCoy, E. 2012. Filtering via approximate Bayesian computation. *Statistics and Computing* 22:1223–1237.

Ko, J., and Fox, D. 2009. GP-BayesFilters: Bayesian filtering using Gaussian process prediction and observation models. *Autonomous Robots* 72(1):75–90.

Ladd, A. M.; Bekris, K. E.; Rudys, A.; Marceau, G.; Kavraki, L. E.; and Wallach, D. S. 2002. Robotics-based location sensing using wireless ethernet. In *Proc. 8th Intern. Conf. on Mobile computing and networking*, 227–238.

Lazebnik, S.; Schmid, C.; and Ponce, J. 2006. Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In *Proc. 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, 2169–2178.

Nishiyama, Y.; Boularias, A.; Gretton, A.; and Fukumizu, K. 2012. Hilbert space embeddings of POMDPs. In *Proc. Conference on Uncertainty in Artificial Intelligence (UAI2012).*

Pronobis, A., and Caputo, B. 2009. COLD: COsy Localization Database. *Intern. J. Robotics Research* 28(5):588–594.

Quigley, M.; Stavens, D.; Coates, A.; and Thrun, S. 2010. Sub-meter indoor localization in unmodified environments with inexpensive sensors. In *Proc. IEEE/RSJ Intern. Conf. on Intelligent Robots and Systems (IROS).*

Rossi, V., and Vila, J.-P. 2009. Convolutional particle filter for parameter estimation in general state-space models. *IEEE Transactions on Aerospace and Electronic Systems* 45:1063–1072.

Schölkopf, B., and Smola, A. J. 2002. *Learning with Kernels.* MIT Press.

Se, S.; Lowe, D. G.; and Little, J. 2001. Vision-based mobile robot localization and mapping using scale-invariant features. In *Proc. Intern. Conf. on Robotics and Automation (ICRA)*, 2051–58.

Smola, A.; Gretton, A.; Song, L.; and Schölkopf, B. 2007. A Hilbert space embedding for distributions. In *Proc. 18th Intern. Conf. on Algorithmic Learning Theory*, 13–31.

Song, L.; Huang, J.; Smola, A.; and Fukumizu, K. 2009. Hilbert space embeddings of conditional distributions with applications to dynamical systems. In *Proc. 26th ICML*, 961–968.

Song, L.; Boots, B.; Siddiqi, S.; Gordon, G.; and Smola, A. 2010. Hilbert space embeddings of hidden Markov models. In *Proc. 27th ICML.*

Song, L.; Gretton, A.; Bickson, D.; Low, Y.; and Guestrin, C. 2011. Kernel belief propagation. In *Proc. 14th Intern. Conf. on Artificial Intelligence and Statistics.*

Song, L.; Fukumizu, K.; and Gretton, A. 2013. Kernel embeddings of conditional distributions. *IEEE Signal Processing Magazine* 30(4):98–111.

Song, L.; Gretton, A.; and Guestrin, C. 2010. Nonparametric tree graphical models. In *Proc. 13th Intern. Conf. on Artificial Intelligence and Statistics*.

Sriperumbudur, B. K.; Gretton, A.; Fukumizu, K.; Schölkopf, B.; and Lanckriet, G. R. 2010. Hilbert space embeddings and metrics on probability measures. *J. Machine Learning Research* 11:1517–1561.

Thrun, S.; Burgard, W.; and Fox, D. 2005. *Probabilistic Robotics.* MIT Press.

Vlassis, N.; Terwijn, B.; and Kröse, B. 2002. Auxiliary particle filter robot localization from high-dimensional sensor observations. In *Proc. Intern. Conf. on Robotics and Automation (ICRA)*, 7–12.