Enhanced Probabilistic Inference Algorithm Using Probabilistic Neural Networks for Learning Control

YANG LI1,2, SHIJIE GUO1,2, (Member, IEEE), LISHUANG ZHU1,2, TOSHIHARU MUKAI3, (Member, IEEE), AND ZHONGXUE GAN4, (Member, IEEE)

1State Key Laboratory of Reliability and Intelligence of Electrical Equipment, Hebei University of Technology, Tianjin 300130, China
2Hebei Key Laboratory of Robot Perception and Human-Robot Interaction, Hebei University of Technology, Tianjin 300130, China
3Faculty of Information Science, Meijyo University, Nagoya 231-0002, Japan
4Institute of AI and Robotics, Fudan University, Shanghai 200433, China

Corresponding author: Shijie Guo (guoshijie@hebut.edu.cn)

This work was supported in part by the National Key Research and Development Plan under Grant 2016YFE0128700, in part by the Natural Science Foundation of Hebei Province under Grant E2017202270, in part by the Key Research and Development Plan of Hebei Province under Grant 18211816D, and in part by the National Key Research and Development Plan under Grant 2017YFB1301002.

ABSTRACT In model-based methods of reinforcement learning (RL), the probabilistic inference for learning control (PILCO) algorithm, which relies on Gaussian process (GP) for building probabilistic dynamics models, is getting attention for its advantages of requiring only a small amount of data, and being able to learn from scratch in a few trials by explicitly incorporating model uncertainty into long-term planning. However, the time complexity of the GP, which is cubic with respect to the number of trials, makes it challenging to scale the framework to high-dimensional observation spaces. Moreover, the cost function of a task is limited as a locally quadratic function for calculating the policy gradient analytically. To solve these problems, we proposed a probabilistic neural network (PNN) to replace the GP in building probabilistic dynamics models and develop a deterministic control policy by using long-term predictions. In particular, we determine model uncertainty through basic prior knowledge and calculate cumulative cost by sampling from state distributions. This approach can help reduce the influence of model error and time consumption. Compared with the state-of-the-art model-based RL, the proposed approach can reconcile data efficiency and speed of learning even in high-dimensional observation spaces.

INDEX TERMS Reinforcement learning, control policy, probabilistic dynamics model, probabilistic neural network.

I. INTRODUCTION

The use of reinforcement learning (RL) for robot control has been gaining popularity over the last few years [1], [2] in a wide spectrum of applications of robotics [3], [4]. Approaches to RL can be divided into two main classes: model-free (also known as direct) and model-based (also known as indirect) methods. The main difference between model-based and model-free learning lies in whether a model of the interactions between the robot and the environment is employed. In model-free methods, e.g., Q-learning [5] and TD-learning [6], there is no model, and thus the rewards and optimal actions are derived by a trial-and-error approach using a physical system. Although this approach has attracted significant interest of scientists and engineers in the field of robot control, sampling trajectories to derive the optimal policy become a disadvantage when applied to robots [7]. The model-based approach is thus preferable.

Model-based methods feature models of transition dynamics that are used to derive the rewards and optimal actions. This characteristic significantly reduces physical interactions between the robot and its environment, and results in significantly less mechanical wear. It can help discover better policies with fewer trials because the dynamics of the model can generalize knowledge of the system to unobserved states [8]. However, the main disadvantage of model-based learning is that it suffers severely from model errors, especially in long-term predictions. In practice, most recent work still relies on...
a large amount of data for the unsupervised training of auto-encoders [9] to build a control policy.

Although available task-specific prior knowledge can reduce model errors and increase data efficiency in RL [10], but a dramatic improvement without prior knowledge can be achieved with probabilistic inference for learning control (PILCO) [11]. PILCO uses the Gaussian Process (GP) to build a probabilistic dynamics model and propagate uncertain state distributions analytically to realize high data efficiency on low-dimensional control tasks. This enables the probabilistic dynamics model to consider the long-term consequences (cumulative cost) of the parameterization of a given control policy [12]. Thus, the key to achieve high data efficiency is building a probabilistic dynamics model [13]–[16] because it can do so by incorporating uncertainty throughout planning and predicting.

Current probabilistic dynamics models rely on GP to propagate state distributions through time. But the time complexity of GP is cubic with respect to the number of trials, makes it challenging to scale the framework to high-dimensional observation spaces. Furthermore, the cost function of a task is limited as a locally quadratic function since the policy gradient needs to be solved analytically. In this paper, we solve these problems by replacing GP with a neural network while maintaining the framework’s probabilistic nature and its data efficiency-related benefits. To this end, we need to incorporate model uncertainty into the long-term planning for model prediction and policy evaluation. Therefore, we propose a probabilistic neural network (PNN) that expresses predictive uncertainty by using basic prior knowledge to build a probabilistic dynamics model. Prior knowledge here is simply a measure of uncertainty, not manual input of knowledge to the model or policy. Policy gradients are obtained by sampling from state distributions; thus, the cost function of a task is no longer limited to local quadratic function. We call the proposed approach as APILCO because it is an approximate scheme over the traditional PILCO. It uses deterministic inference for long-term prediction and policy evaluation. The framework is similar to that of PILCO but reduces computation time by using PNN instead of GP, and deals with arbitrary cost function by sampling from state distributions.

The paper is organized as follows: after discussing related work in Section II, we describe the key ideas of the proposed APILCO learning framework in Section III, i.e., the probabilistic dynamics model, the approximate cumulative cost evaluation, and the approximate gradient-based policy improvement. Experimental results are provided in Section IV, where we discuss the key properties, limitations, and future extensions of APILCO. Conclusions are drawn in Section V.

## II. RELATED WORK

Ways of treating parameter uncertainty in a control system have long been a key subject of research in robust and adaptive control [17]. A certainty equivalence approach is usually applied, which takes the estimated values of model parameters as true values without uncertainty [18]. PILCO, a model-based RL algorithm, builds policy by taking model uncertainties as noise [19]. It explicitly considers uncertainties to train a control policy to achieve unprecedented data efficiency on several control benchmarks [20]. In this section, we outline PILCO and analyze the key areas where it is effective.

The parameters of a deterministic policy $\pi$ are initialized with random parameters $\theta$; PILCO then executes the policy from initial states (sampled from an initial distribution $p(S_0)$) until time horizon $T$. During policy execution, the observed transitions are recorded and appended to the whole training data. The dynamics model is then re-trained by the GP model using all transition data. For a certain state $s_{t-1}$, PILCO can predict a Gaussian distribution of the next state $p(s_t|s_{t-1}, a_{t-1}) = N(\mu_t, \Sigma_t)$, and this is consistent with transitions of the Markov model. According to this, PILCO can analytically predict subsequent state distributions from an initial state distribution (e.g., $p(S_0)$ to $p(S_T)$) and state distributions are correspondent to a joint Gaussian assumption $s_t \sim N(\mu_t, \Sigma_t)$. These distributions can evaluate the cumulative cost of policy $\pi$

$$J_\pi(\theta) = \sum_{t=0}^{T} E [c(s_t)],$$

where both $\mu_t$ and $\Sigma_t$ are functionally dependent on $\mu_t, \Sigma_t$. Hence, PILCO can analytically compute the gradients of cumulative cost $J_\pi$ with respect to policy parameters $\theta$. The cumulative cost $J_\pi$ is finally optimized to yield an improvement in policy. PILCO then executes the optimized policy in a new trial to determine whether it is locally optimal given the probabilistic dynamics model.

The data efficiency of PILCO can be attributed to its probabilistic dynamics model that avoids model bias. Model bias arises from the selection of a single dynamics model $\hat{f}$, which is assumed the correct model without uncertainty [15]. While a single dynamics model can provide accurate short-term
state predictions (e.g., \( p(S_1) \)), its long-term predictions (e.g., \( p(S_T) \)) are inaccurate due to the compounding effects of errors in T-step prediction. Because optimizing the cumulative cost \( J_e \) must balance the perceived cost per distributions, e.g., \( p(S_1) \) and \( p(S_T) \), the optimization compromises on the cost of \( p(S_1) \) based on the perceived cost of \( p(S_T) \), though the prediction \( p(S_T) \) is almost completely random noise, i.e., given sufficient model uncertainty, \( p(S_T) \) has a wide distribution that is nearly invariant to policy \( \pi \). PILCO avoids model error through a probabilistic dynamics model that can express uncertainty, and integrates model uncertainties into long-term planning and decision-making.

### III. APPROXIMATE PROBABILITY INFERENCE

In the following, we detail the key components of APIILCO: the probabilistic dynamics model, the approximate cumulative cost evaluation, and the approximate gradient-based policy improvement.

#### A. PROBABILISTIC DYNAMICS MODEL LEARNING

We do not use GP to build the probabilistic dynamics model (PDM), and instead propose PNN to estimate the variance of state distributions on which we condition our inference. With PNN as a parametric model, the parameter set imposes a fixed structure upon the prediction. The number of parameters is determined in advance, and is independent of the number of observations and sample size. PNN makes one-step predictions as a Gauss distribution that is established by using the predict values as means, and the uncertainties as variances.

We assume the tasks are within a finite state space and normalize the states before training. The input to the PDM is \( \tilde{x}_{t-1} = (s_{t-1}, a_{t-1}) \). It performs one-step prediction to yield the mean \( \mu_t \) and the appended output \( sa = [se, ss] \), which is used to calculate the variance \( \Sigma_t \) of the predict Gaussian distribution.

The training target of \( \mu_t, sa_t \) is \( s_t^* \in \mathcal{R}^D, sa_t^* = \alpha \times [\cos(s_t^*), \sin(s_t^*)] \in \mathcal{R}^{2D} \), where \( \alpha \) is a hyper-parameter constraining \( sa_t \) to \([-\alpha, \alpha]\) and bounded by the use of tanh as the activation function of the last layer of the PDM.

\[
\begin{align*}
\{\tilde{\mu}_t, sa_t\} &= f(\tilde{x}_{t-1}).
\end{align*}
\]

The training target of the PDM is \( y_t^* = [s_t^*, sa_t^*] \in \mathcal{R}^{2D} \). By the trigonometric function, we have a basic priori knowledge

\[
1 - \cos^2(s_t^*) - \sin^2(s_t^*) = 0.
\]

The entire training set is \( \mathcal{D} = \{\tilde{x}_i = [\tilde{x}_0, \ldots, \tilde{x}_{n-1}], y_i = [y_1^*, \ldots, y_n^*]\}, \) assuming that the training errors of the PDM on set \( \mathcal{D} \) are

\[
(f(\tilde{x}) - Y)^2 < \varepsilon.
\]

In addition, we define the evaluation of uncertainty as

\[
\text{Eva}(sa_t) = 1 - se_t^2 - ss_t^2 < 2\varepsilon.
\]

Our prediction of variance \( \Sigma_t \) is based on a higher-level assumption that there is a correlation between the precision of prediction and the difference between \( f(\tilde{X}) \) and \( Y \). Therefore, there is also a correlation between predictive uncertainties and \( \text{Eva}(sa_t) \). We express this uncertainty by the variance \( \Sigma_t \) of the Gaussian distribution as

\[
\Sigma_t = 1 + e - \exp\left( -\frac{\text{Eva}(sa_t)}{\beta} \right),
\]

where \( e \) and \( \beta \) are hyper-parameters. Thus, the Gaussian distribution can be obtained as \( s_t \sim \mathcal{N}(\tilde{\mu}_t, \tilde{\Sigma}_t) \). Unlike the GP model that becomes ineffective in high-dimensional space, the PNN is good at processing high-dimensional data, and this is a significant advantage for PILCO.

We finally tested the effectiveness of our prediction and evaluated the uncertainty of PNN and GP with small sample data. As shown in Fig. 1, PNN achieved almost the same results with GP, though PNN is a parametric model.

#### B. POLICY EVALUATION

Minimizing and evaluating \( J_e \) in (1) requires long-term predictions of the evolution of state. The one-step predictions of the PDM are obtained as shown in (2) and (6), analytical solutions of \( f(\tilde{x}) \) form the premise to obtain those of the state distributions at each step \( p(S_1), \ldots, p(S_T) \).

To predict \( s_{t-1} \) from \( p(S_{t-1}) \), we require a distribution \( p(s_{t-1}, a_{t-1}) \), as the policy \( \pi \) is deterministic, which means \( a_{t-1} = \pi(s_{t-1}, \theta) \). Both one-step predictions \( s_t \) are correspondent with the Gaussian distribution, the process of prediction of the PDM is given as a black box. Therefore, we compute the cumulative cost per step through sampling from state distributions. At time \( t \), the mean and variance of the distribution of the output of each sample \( i \) are \( \tilde{\mu}_t^i \) and \( \tilde{\Sigma}_t^i \), respectively, where \( i \in [1, \ldots, M] \). Thus, the mean \( \mu_t \) of the predicted state distribution can be approximated as

\[
\mu_t = E(s_t) = \frac{1}{m} \sum_{i=0}^{m} \tilde{\mu}_t^i.
\]

To compute the covariance matrix \( \Sigma_t \) of the state distribution, we need to distinguish diagonal and off-diagonal elements. Using the law of iterated variances, we obtain the target dimensions \( i, j \in [1, \ldots, D] \)

\[
\begin{align*}
\sigma_{ii}^2 &= E_{s_{t-1}} \left[ (\tilde{\Sigma}_t^i)^{1/2} \right] + E_{s_{t-1}} \left[ (\tilde{\mu}_t^i)^{1/2} \right] - (\tilde{\mu}_t^i)^2, \\
\sigma_{ij}^2 &= E_{s_{t-1}} \left[ \tilde{\mu}_t^i \tilde{\mu}_t^j \right] - \tilde{\mu}_t^i \tilde{\mu}_t^j, \quad i \neq j,
\end{align*}
\]

where \( \tilde{\mu} \) is known from (2) and \( \tilde{\Sigma} \) is known from (6). The off-diagonal terms do not contain additional term \( E(\tilde{\Sigma}_t^i) \) because our approach follows the assumption of conditional independence of GP models: Different target dimensions do not covary for a given \( x_{t-1} \). The analytical solution can be expressed as

\[
\begin{align*}
E_{s_{t-1}} \left[ \tilde{\Sigma}_t \right] &= \int \tilde{\Sigma}_t N(s_{t-1}; \tilde{\mu}_{t-1}, \tilde{\Sigma}_{t-1}) \\
E_{s_{t-1}} \left[ (\tilde{\mu}_t)^2 \right] &= \int (\tilde{\mu}_t)^2 N(s_{t-1}; \tilde{\mu}_{t-1}, \tilde{\Sigma}_{t-1})
\end{align*}
\]
In our implementation, we calculate (10) on a set of \( M \) candidate values sampled randomly from a Gaussian distribution \( p(S_{t-1}) \), which provides suitably diverse samples to approximately solve the equations

\[
\begin{align*}
E_{S_{t-1}} \left[ \tilde{\Sigma}_t \right] & \approx \sum_{i=0}^{m} \tilde{\Sigma}_i^i \\
E_{S_{t-1}} \left[ (\tilde{\mu}_t)^2 \right] & \approx \sum_{i=0}^{m} (\tilde{\mu}_i)^2
\end{align*}
\]

such that it cannot obtain the solution for \( p(S_t) \sim \mathcal{N}(\mu_t, \Sigma_t) \). Although the state distributions are approximated by the samples, they exhibit a few advantages and there are no restrictions on policy \( \pi \). We can implement a policy of infinite complexity. The cumulative cost is

\[
J_\pi (\theta) = \sum_{t=0}^{T} \int c(s_t) \mathcal{N}(s_t|\mu_t, \Sigma_t) \, ds_t. \tag{12}
\]

If the integration required for (12) is analytically intractable due to the existence of \( c(s_t) \), \( J_\pi \) can be approximated as

\[
J_\pi (\theta) \approx \sum_{t=0}^{T} \sum_{i=0}^{m} c(s_t). \tag{13}
\]

### IV. EXPERIMENT AND DISCUSSION

This section describes the experimental setup. All tasks were in a continuous state and involved continuous action in discrete time to test the validity of the proposed algorithm. To intuitively show how the algorithms work, we present a detailed analysis of the key properties of APILCO.

#### A. TASKS FOR COMPARISON

In the following, we evaluate the quality of APILCO for long-term predictions in terms of computational requirements and learning speed. Moreover, we reveal the properties of the PNN. For these assessments, we apply APILCO to nonlinear control tasks and compare it with PILCO.

1) TASK DESCRIPTION

We considered two numerical simulation tasks, i.e., double-pendulum swing up and cart pole swing up, to evaluate the validity of the APILCO: Learning speed and quality of approximate inference. More details of the tasks will be provided in the following sections.

Double-pendulum swing-up task: The double-pendulum system is a two-link robot arm with two actuators, as shown in Fig. 2(a). The state \( s \) is given by the angles \( \theta_1 \) and \( \theta_2 \). The corresponding angular velocities \( \dot{\theta}_1 \) and \( \dot{\theta}_2 \) of the two links are measured with respect to the vertical. Each link is 1 m in length and weighs 0.5 kg. The torques \( u_1 \) and \( u_2 \) are in the range from \(-3\) to \(3\) Nm. The control signals are changed with an interval of 50 ms. The objective is to learn a control policy that can swing the double pendulum up from an initial distribution \( \rho(s_0) \) around \( \mu_0 = [\pi, \pi, 0, 0]^T \) and balance it to the inverted position with \( s_d = [0, 0, *, *] \), where * represents arbitrary velocity due to the cost function is independent of velocities. The prediction period is within 1.5 s.
The task is challenging because it is required to obtain the interplay of two correlated control signals. To solve the double-pendulum swing-up task, we parametrized the preliminary policy as a multilayer neural network herein.

Cart pole swing-up task: The cart pole system consists of a cart moving along the horizontal and a freely swinging pendulum attached to it, as shown in Fig. 2 (b). The state of the system is expressed with the position $x$ of the cart, the velocity $\dot{x}$, the angle $\theta$ (the pendulum measured hanging downward), and the angular velocity $\dot{\theta}$. In our simulation, we set the masses of the cart and the pendulum to 0.700 kg and 0.325 kg, respectively, the length of the pendulum to 0.6 m, and the coefficient of friction between cart and ground to 0.1 Ns/m. The prediction period is set to 2.5 s. The control signal is changed with an interval of 100 ms.

A horizontal force $u \in [-10, 10]$ N is applied to the cart. The objective is to learn a control policy that can swing the pendulum up from around $\mu_0 = [x, \dot{x}, \theta, \dot{\theta}]^T$ and balance it to the inverted position while keeping the cart in the middle, i.e., $s_d = [0, s, \pi, s]^T$. A linear policy cannot be used to solve the task [21]; APILCO learns a nonlinear policy with the multilayer neural network.

To compare it with PILCO, for both tasks, we use a cost function as PILCO requires that

$$c(s) = 1 - \exp\left(-\frac{\|s - s_d\|^2}{\sigma_c^2}\right), \quad (14)$$

where $\sigma_c = 0.5$. The Euclidean distance between the target position and the tip of the pendulum is penalized.

2) RESULTS AND DISCUSSION

In this section, we compare the results of APILCO and PILCO on the tasks. Fig. 2 shows the results. The architecture of the PNN had 150 units with three hidden layers and sigmoid activation functions to fit the dynamics model. The control policy was a three-layer neural network with 150 units, and the weight decay of the NN dynamics model was set to $2 \times 10^{-3}$. We used a “replay buffer” with finite size (the most recent 10 episodes) and discarded older episodes of data.

We alternated between fitting a dynamics model and optimizing the control policy. We first generated a single random episode when experimenting with each method before iterating over the main loop for 50 episodes. In each iteration, a single episode of data was acquired by executing tasks for the prediction horizon and generating a transition per episode. We evaluated each method in each iteration by calculating the total cost and time consumption. Both PILCO and APILCO used the same physical simulator and cost function settings.

Fig. 3 shows the total costs of PILCO and APILCO. In both tasks, PILCO converged after 7-8 episodes whereas APILCO required 10–20 episodes before reaching a cost slightly lower than that incurred by PILCO. Although APILCO was not as data efficient as PILCO, it was considerably efficient in terms of time consumption as shown in Fig. 4.

We counted the running time of both models for both APILCO and PILCO, where both algorithms leveraged the CPU. The time consumption of APILCO for 50 episodes was only 16 minutes. Whereas PILCO’s simulation by contrast took 250 minutes for 50 episodes on the same computer.

The time complexity of the simulation (with gradient information) was technically constant. Our experiments used a dataset of size $N$, input dimensionality $X$, and action dimensionality $U$. Because the dynamics model was trained only on the 10 most recent episodes (so that the PNN would assign a larger weight to newer information more likely useful for training a control policy than older data), we used P-many samples to represent the distributions.

However, because the “replay buffer” was limited to ten recent trials, the time complexity of APILCO was constrained to $O(PN(X + U))$ while PILCO had a time complexity of $O(N^2X^2(X + U)^2)$ [22]. With more data collected, PILCO took more time for each trial. The minimum time for convergence of PILCO was beyond the requirement of APILCO for both tasks. Consequently, PILCO is unsuited to tasks requiring a large number of episodes or those with high-dimensional states.
Fig. 5 shows the total cost and cumulative cost of APILCO per trial, where the cumulative cost in the PDM was slightly higher than the total cost in a dynamics model. It causes the fact that the PDM slightly overestimated the variance of the state distributions.

Fig. 6 shows a typical example of the position of the car and the angle of the pendulum. In the early episode of learning, PDM is not ideal for the long-term prediction of state distribution. The distributions of the states were multimodal. APILCO dealt with this inappropriate modeling by learning a control policy that forced the states into a unimodal distribution such that a Gaussian approximation was appropriate, as shown in Figs. 6(c) and (d).

We explain this behavior as follows: Assuming that APILCO found different paths that led to a target, a wide Gaussian distribution was required to capture all possible paths.

However, when computing the expected cost, uncertainty in the predicted state led to higher expected cost, assuming that the mean was close to that of the target. Therefore, APILCO optimized the control policy to push the marginally multimodal trajectory distribution into a single mode.

B. SCALING TO HIGHER DIMENSIONS OR/AND ARBITRARY COST FUNCTION TASKS

To implement the calculation of the analytical gradient, PILCO needed to use the saturating cost function in (15). However, the cost function directly affects policy research, and is usually determined by the tasks rather than the algorithms [23]. APILCO calculates the cumulative cost by
sampling from state distributions, as in (14). Therefore, APILCO can deal with tasks with an arbitrary cost function. For these assessments, we applied it to three nonlinear control tasks.

1) TASK DESCRIPTION

We considered three simulated tasks (cart pole swing up, two-DOF manipulator location task, and seven-DOF manipulator location task). The cost functions were determined by tasks rather than algorithms.

The state of the cart pole system was described by position \( x \), velocity \( \dot{x} \), complex representation \([\cos(\theta), \sin(\theta)]\) of angle \( \theta \) of the pendulum measured when hanging downward, and angular velocity \( \dot{\theta} \). The objective was to learn a control policy to swing the pendulum up from around \( \theta_0 \) and \( \dot{\theta}_0 \) to approximately \( \frac{\pi}{2} \), which was the target angle and rotating velocity \( \dot{\theta}_d \). The prediction horizon was 10 s. The simulation model is given in Fig. 7 (a).

We chose the cost function as a function of Euclidean distance and rotating velocity

\[
c(s) = \frac{1}{2}(s(s) - s_d)^2 + \frac{1}{2}(\dot{\theta})^2 + |a|^2,
\]

where \( d \) is the Euclidean distance between the target position and the end position of the manipulator, and \( \alpha = [\alpha_1, \alpha_2]^T \) is a constant weight vector.

The seven-DOF manipulator consisted of seven links as shown in Fig. 7 (c) The state \( s \) was given by the complex representation of angles \([\cos(\theta_1), \sin(\theta_1), \ldots, \theta_7]\), and the corresponding angular velocities \( \dot{\theta}_i \) of the links. The control signals were changed every 20 ms. The objective was to learn a control policy that forced the links from an initial distribution \( p(x_0) \) and kept the end of the manipulator in the target state \( s_d = [\theta_d, \dot{\theta}_d] \) (where \( \theta_d \) is the target angle and \( \dot{\theta}_d \) is the target velocity set to as zeros). The prediction horizon was 2 s.

We chose the cost function as

\[
c(x) = \beta(\theta - \theta_d)^2 + \frac{1}{2}|\dot{\theta}|^2 + |a|^2,
\]

where \( \beta = [\beta_1, \beta_2, \ldots, \beta_7]^T \) is a weight vector, and \( a \) is the torque applied to each link.

Note that the angle of the tasks used complex representation \([\cos(\theta_1), \sin(\theta_1)]\), which were correlated variables. However, the state variables were independent random variables in the assumption of the PDM. The influence of this is discussed later.

2) EVALUATION OF KEY PROPERTIES

In this part, the key properties of the approximate method of inference in terms of the influence of independence on state distribution and that of initial distribution on APILCO are discussed.

We compared the total cost and cumulative cost per trial in Fig. 8. APILCO maintained excellent performance on tasks.
FIGURE 6. Samples of long-term predictive (Gaussian) distributions for the predicted (scatter) and real samples (solid line). (a)/(b) are the real and predicted positions/angles in the early stages of learning, and the Gaussian approximation of PNN is a suboptimal choice. (c)/(d) are the real and predicted positions/angles when APILCO learned a feasible control policy, the Gaussian approximations of the predictive state distributions are consistent with the real trajectory, which is also consistent with the real trajectory given by the learned control policy of PILCO.

FIGURE 7. Examples of tasks solved using APILCO: (a), cart pole task, (b), two-DOF manipulator location task, and (c), seven-DOF manipulator location task. The cost functions are determined by each task rather than the algorithms.

with arbitrary cost functions, where these tasks could not be implemented by PILCO due to the limitations of its cost function. Although we could not directly compare the cumulative cost obtained by the analytical solution in (13) with that obtained by sampling approximation in (14), the learning success of the tasks proved that sampling was effective in approaching analytical cumulative cost.

It was assumed that state variables of the PNN were independent in the three tasks mentioned above. Although, the angles were given by the complex representations (sine and
FIGURE 8. Total and cumulative costs of each task of APILCO per trial: (a), the cart pole task, (b), the two-DOF manipulator location task, and (c), the seven-DOF manipulator location task.

FIGURE 9. Samples of long-term predictive (Gaussian) distributions during predicted (scatter) and real trajectory (solid line) in the 58th trial. The sine and cosine values of the angles are given by (a) and (b), respectively. Although a high variance in the distribution of predictive states was caused by a random initial state, the real trajectory maintained impressive performance.

cosine functions) which violated the assumption of the independence of variables, APILCO was still effective for these tasks.

In case the variables became inconsistent with the correlation owing to the complex representation, a difference between the predicted states and real states would be
generated by a larger variance in the distributions of the predicted states. Larger variance was contrary to low cumulative cost. However, the target of the improvement in control policy was to reduce the cumulative cost, which forced the variables of the predictive states to be correlative variables.

As shown in Fig. 8, in tasks (a) and (c), the correlative of the variables had no negative effect. In task (b), although it had no effect on total cost, it rendered the cumulative cost unstable. The difference between task (b) and the others was that the initial state \( x_0 \) was randomly chosen as the mean of the initial distribution to improve control policy. We chose the 58th episode in the learning for control policy as an example to illustrate the phenomenon. The cumulative cost of the episode was overestimated as shown in Fig. 8 (b).

The overestimation of cumulative cost can be explained intuitively. In the trial, variables of subsequent states were always the correct correlative variables even if the initial state was unobserved. When the subsequent states fitted the predictions, the control policy could successfully complete the task as shown in Fig. 9. Nevertheless, in long-term predictions, an unobserved state might have led to the variables of subsequent states inconsistent with the correlation; while a wide Gaussian distribution was required to capture the variability of the state distribution in this case, which was almost invariant with regard to policy \( \pi \).

V. CONCLUSION
To overcome the problems in the current PILCO algorithm, we replaced GP by PNN in building probabilistic dynamics models and developed a deterministic control policy through long-term predictions. We determined model uncertainty through basic prior knowledge and calculated cumulative cost by sampling from state distributions. Simulation experiment demonstrated that the proposed approach could expand the probabilistic inference framework to high-dimensional observation spaces. It had a high learning speed with similar data efficiency to PILCO, and does not require human input, as well as other kinds of informative initializations or prior knowledge. In addition, PNN could avoid the cubic time complexity of GP and overcome the restriction on cost function. It was shown that PNN was useful for learning probabilistic dynamics models and formulating a control policy.

REFERENCES
[1] V. Mnih, David Silver, Andrei A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kuban, D. Wierstra, S. Legg, and D. Hassabis, “Human-level control through deep reinforcement learning,” Nature, vol. 518, no. 7540, pp. 529, 2015.
[2] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Diehelm, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, “Mastering the game of Go with deep neural networks and tree search,” Nature, vol. 529, no. 7587, pp. 484–489, 2016.
[3] M. Jaderberg, V. Mnih, W. M. Cranke, T. Schaul, J. Z. Leibo, D. Silver, and K. Kavukcuoglu, “Reinforcement learning with unsupervised auxiliary tasks,” Accessed: Nov. 16, 2016. [Online]. Available: https://arxiv.org/abs/1611.05397
[4] C. G. Atkeson and J. C. Santamaria, “A comparison of direct and model-based reinforcement learning,” in Proc. Int. Conf. Robot. Automat., Albuquerque, NM, USA, Apr. 1997, pp. 3557–3564.
[5] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, “Playing Atari with deep reinforcement learning,” Accessed: Dec. 19, 2013. [Online]. Available: https://arxiv.org/abs/1312.5602
[6] J. N. Tsitsiklis and B. Van Roy, “An analysis of temporal-difference learning with function approximation,” IEEE Trans. Autom. Control, vol. 42, no. 5, pp. 674–690, May 1997.
[7] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, D. Wierstra, “Continuous control with deep reinforcement learning,” Accessed: Sep. 9, 2015. [Online]. Available: https://arxiv.org/abs/1509.02971
[8] Y. Gal, R. McAllister, and C. E. Rasmussen, “Improving PILCO with Bayesian neural dynamic models,” in Int. Conf. Mach. Learn., New York, NY, USA, 2016.
[9] B. C. Stadie, S. Levine, and P. Abbeel, “Incentivizing exploration in reinforcement learning with deep predictive models,” Accessed: Jul. 3, 2015. [Online]. Available: https://arxiv.org/abs/1507.00814
[10] S. Schaal and C. G. Atkeson, “Learning from demonstration,” Rob. Autom. Syst., vol. 47, no. 2, pp. 65–67, 2004.
[11] P. Dayan and Y. Niv, “Reinforcement learning: The good, the bad and the ugly,” Curr. Opin. Neurobiol., vol. 18, no. 2, pp. 185–196, 2008.
[12] D. Deisenroth, D. Fox, and C. E. Rasmussen, “Gaussian processes for data-efficient learning in robotics and control,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 37, no. 2, pp. 408–423, Feb. 2015.
[13] M. P. Deisenroth, C. E. Rasmussen, and J. Peters, “Gaussian process dynamic programming,” Neurocomputing, vol. 72, nos. 7–9, pp. 1508–1524, 2009.
[14] Y. Engel, S. Mannor, and R. Meir, “Bayes meets Bellman: The Gaussian process approach to temporal difference learning,” in Proc. Int. Conf. Mach. Learn., Washington, DC, USA, 2003, pp. 154–161.
[15] C. E. Rasmussen and M. Kuss, “Gaussian processes in reinforcement learning,” in Proc. Adv. Neural Inf. Process. Syst., Vancouver, BC, Canada, 2004, pp. 751–758.
[16] A. Wilson, A. Fern, and P. Tadepalli, “Incorporating domain models into Bayesian optimization for RL,” in Machine Learning and Knowledge Discovery in Databases. Berlin, Germany: Springer2010, pp. 467–482.
[17] X. Guo, W. Yan, and R. Cui, “Event-triggered reinforcement learning-based adaptive tracking control for completely unknown continuous-time nonlinear systems,” IEEE Trans. Cybern., to be published.
[18] M. R. James, “On the certainty equivalence principle and the optimal control of partially observed dynamic games,” IEEE Trans. Autom. Control, vol. 39, no. 11, pp. 2321–2324, Nov. 1994.
[19] J. G. Schneider, “Exploiting model uncertainty estimates for safe dynamic control learning,” in Proc. Adv. Neural Inf. Process. Syst., 1997, pp. 1047–1053.
[20] M. Deisenroth and C. E. Rasmussen, “PILCO: A model-based and data-efficient approach to policy search,” in Proc. Int. Conf. Mach. Learn., Washington, DC, USA, 2011, pp. 465–472.
[21] T. Raiko and M. Tornio, “Variational Bayesian learning of nonlinear hidden state-space models for model predictive control,” Neurocomputing, vol. 72, nos. 16–18, pp. 3702–3712, 2009.
[22] R. McAllister, “Bayesian learning for data-efficient control,” Ph.D. dissertation, Dept. Eng., Cambridge Univ., Cambridge, MA, USA, 2017.
[23] D. Dewey, “Reinforcement learning and the reward engineering principle,” in AAAI Spring Symp., San Francisco, California, USA, 2014.
SHIJIE GUO received the Dr.Eng. degree in mechanical engineering from the Tokyo Institute of Technology, in 1992. He was a Team Leader with the Robot Implementation Research Team, RIKEN-TRI Collaboration Center for Human-Interactive Robot Research, RIKEN, and the Head of the Healthcare Division of Sumitomo Riko Ltd., Japan, from August 2007 to March 2015. He is currently a Professor with the School of Mechanical Engineering and the Head of the Hebei Key Laboratory of Robot Perception and Human-Robot Interaction, Hebei University of Technology, Tianjin, China. He also serves as a Professor with the Institute of AI and Robotics, Fudan University, and the Principal Scientist with the State Key Laboratory for Reliability and Intelligence of Electrical Equipment. His current research interests are nursing-care systems including transfer assistant robots, walking assist robots, smart care beds, and unconstrained measurement of vital information. His is a member of JSME. He also serves as the Editor-in-Chief of the Journal of Hebei University of Technology.

LISHUANG ZHU received the B.S. degree from the Harbin Institute of Technology, China, in 2017. She is currently pursuing the M.S. degree with the Hebei University of Technology. Her main research interests include control of manipulators as well as machine learning and its application to the motion control of robots.

TOSHIHARU MUKAI received the Dr.Eng. degree in information physics from The University of Tokyo, in 1995. He was a Frontier Research Scientist with the Bio-Mimetic Control Research Center, RIKEN, Japan, from April 1995 to September 2000. From October 2000 to September 2001, he stayed at the Laboratoire de Neurobiologie, Marseille, France, as a Postdoctoral Fellow. He was the Laboratory Head of the Biologically Integrative Sensors Laboratory, Bio-Mimetic Control Research Center, RIKEN, from October 2001 to September 2008. From August 2007 to March 2015, he was a Team Leader of the Robot Sensor Systems Research Team, RIKEN-TRI Collaboration Center for Human-Interactive Robot Research. Since April 2015, he has been a Professor at the Department of Information Engineering, Faculty of Science and Technology, Meijo University, Japan. His current research interests include nursing-care assistant robots, robotic sensors, and sensor applications to welfare apparatus. He is a member of JSR.

ZHONGXUE GAN received the Dr.Eng. degree in mechanical engineering from the University of Connecticut, USA, in 1993. He was a Research Fellow at ABB Group, before he was back to China, in 2006. He is currently the Dean of the Institute of AI and Robotics, Fudan University, China. He is a Key Member of the Transformative Technology Committee of China. He has completed many national key research and development projects on intelligent robots, such as the 863 Program, the National Key Research and Development Project of China. He served as the Chairman of the 3rd, 4th, 5th, and 6th China Robot Summit. His research interests include distributed intelligent systems, path planning, and control of industrial robots.

***