Latent Representation in Human–Robot Interaction With Explicit Consideration of Periodic Dynamics

Taisuke Kobayashi, Member, IEEE, Shingo Murata, Member, IEEE, and Tetsunari Inamura, Member, IEEE

Abstract—This article presents a new data-driven framework for analyzing periodic physical human–robot interaction (pHRI) in latent state space. The model representing pHRI is critical for elaborating human understanding and/or robot control during pHRI. Recent advancements in deep learning technology would allow us to train such a model on a dataset collected from the actual pHRI. Our framework is based on a variational recurrent neural network (VRNN), which can process time-series data generated by a pHRI. This study modifies VRNN to explicitly integrate the latent dynamics from robot to human and to distinguish it from a human state estimate module. Furthermore, to analyze periodic motions, such as walking, we integrate VRNN with a new recurrent network based on reservoir computing (RC), which has random and fixed connections between numerous neurons. By boosting RC into a complex domain, periodic behavior can be represented as phase rotation in the complex domain without decaying the amplitude. A rope rotation/swinging experiment was used to validate the proposed framework. The proposed framework, trained on the collected experiment dataset, achieved the latent state space in which variation in periodic motions can be distinguished. The best prediction accuracy of the human observations and robot actions was obtained in such a well-distinguished space.

Index Terms—Complex domain, human–robot interaction, latent space extraction, motion analysis, recurrent neural networks (RNNs).

I. INTRODUCTION

As the birthrate declines and the population ages, robots to supplement human labor are getting highly desired. Such robots are required to perform tasks in uncertain real-world situations, particularly those involving physical human–robot interaction (so-called pHRI) [1], such as in the welfare and service industries rather than tasks in predetermined environments, such as factory automation. A growing trend toward the development of practical pHRI-capable robots stimulates the pHRI research: for example, because simulation is essential for the development of robots, a simulation platform that can handle pHRI has been developed and served [2], [3]; while human understanding through pHRI (by robot collecting data about the interaction with a human) is an open and challenging issue [4], [5].

A virtual impedance model is frequently used to control the robot’s physical interaction [6], [7]. It allows the robot to readily follow humans while maintaining stability, but the contact locations and activities that can be generated are severely limited. Even if the impedance parameter is altered adaptively, as shown in the preceding literature and previous studies using learning-based estimation of human motion intention [8], [9], these limitations are imposed by the model and cannot be solved essentially. Such limitations may be alleviated by incorporating add-on mechanisms, such as a barrier function [10], but doing so would inevitably complicate the entire pHRI system and impair its original ease of use. Although research on robots having tactile sensors all over the body has been conducted [11], [12], their controller is still built on a pre-assumption model and does not contain any particular human characteristics. For the development of increasingly complex controllers, general yet practical pHRI model is required.

Recent remarkable advancements in deep learning technology [13] will shed light on the modeling of pHRI from a dataset derived from the actual interactions. Methods for extracting the latent state space hidden in high-dimensional observation data have been established as variational autoencoder (VAE) [14], [15]. They have been applied to control applications based on learning of dynamics model [16], [17] and motion classification in the extracted space [8]. Following these previous studies, we focus on the analysis of pHRI in latent space.

When considering pHRI on latent space, it is important to remember that it generates time-series data. In the previous studies, therefore, recurrent neural networks (RNNs) (e.g., [18], [19]) are often included in the network architecture to extract time-series features into the latent state [8], [20], [21]. Additionally, dynamics (i.e., state transitions in response to a robot action) are explicitly included in the latent space for generating time-series behaviors [21], [22].
When combining these two modules, they should be properly separated. In particular, the dynamics module must satisfy the Markov property (i.e., the next state is only determined by the current state and the action at that time) from a perspective of control theory [23], [24]. However, all of the methods introduced above included RNNs after the dynamics. In this case, the RNNs would complement and/or overwrite the time-series features, removing the requirements for the features input to the dynamics to satisfy the Markov property. The resulting dynamics may not be suitable for control applications.

Furthermore, when formulating the VAE optimization problem, the extraction of latent states for time-series data should be considered. Although it lacks dynamics in the latent state space, a variational recurrent neural network (VRNN) would be a promising method for integrating VAE with RNNs [25]. That is, we need to reformulate the original VRNN in a new formulation that explicitly includes the latent dynamics.

The final issue is caused by RNNs. While RNNs and their variants are indeed suitable for representing time-series data, they need intractable computational resources when being applied to long time-series data with ambiguous segmentation owing to backpropagation through time (BPTT) [26]. Although truncation is commonly employed in implementations to keep the training cost constant, it has been reported to cause bias [27]. Furthermore, many RNNs have been developed with a focus on long-term memory capacity like long short-term memory (LSTM) [18], however, it is doubtful whether pHRI requires such capability. In reality, most pHRI tasks require periodicity rather than large time delays, such as, polishing [6], brushing [7], walking [8], dancing [11], and so on.

From the abovementioned, the issues in the original VRNN (and RNNs) can be summarized as follows.

1) Separation of the latent state estimator and latent dynamics in VRNN for handling time-series data.
2) Reducing the training cost of RNNs.
3) Making RNNs highly representable for periodic time-series data.

To address issue 1), we propose a new VRNN framework for modeling pHRI (see Fig. 1). To differentiate the state estimator from the dynamics, our framework simplifies the input–output relationship in the original VRNN and explicitly incorporates latent dynamics into it. This extension produces a controllable latent human state that can be optimized using a new variational lower bound. Furthermore, in anticipation of future applications in robot control based on this framework, a basic example of auxiliary optimization problems for the robot action policy, i.e., behavioral cloning [23], is introduced.

In this framework, to resolve issue 2), we employ reservoir computing (RC) [28]–[31], a type of RNNs that consist of numerous neurons connected by random and fixed parameters. RC does not require BPTT with fixed parameters, and the representation capability for time-series data is ensured by random and numerous neuron connections.

For 3) inherently representing periodicity, although the parameters and internal states of neurons in RC are originally given in the real domain, we augment them to complex domain [32], [33], so-called complex-valued RC or CRC. The real-valued output of CRC exhibits naturally periodic behavior due to phase rotation in the complex domain without diminishing the amplitude. This natural property contributes to the proposed framework’s ability to represent the periodic pHRI.

A rope-rotation/swinging experiment, in which a human and a direct-drive actuation system collaborate forcefully through a rope to rotate or swing it, is analyzed to validate the proposed framework. Its dataset contains eight different types of motions: rotation or swinging, with slow or fast speed, and by the left or right arm. This dataset is used to train the proposed framework to generate new but similar trajectories to those in the dataset. As a result, even though no label information for the dataset’s motions is provided, the proposed framework successfully constructs the eight clusters corresponding to the dataset’s motions in the extracted latent state space. The proposed framework delivers the maximum prediction accuracy for human observations and robot actions with such a well-distinguished latent state space, whereas performance declines when either of the important components in the proposed method is missing.

II. PROPOSED FRAMEWORK

A. Overview

The typical approaches in the conventional pHRI are either predesign of the modules involved and system integration like [11], or end-to-end learning as a single system like [34]. Both approaches have advantages and disadvantages: the modular structure allows for easy incorporation of prior knowledge of pHRI into the respective modules and to respond flexibly to system changes by switching the corresponding modules, while adaptability through learning allows for dealing with situations that cannot be assumed by predesign based on experiences. The framework of this research aims to develop a new learnable modular structure that inherits these advantages.

The pHRI model, in instance, is believed in our framework to be capable of revealing the human state from high-dimensional observation, though the method for doing so is unknown. Therefore, the current observation and the history of interaction should be used to evaluate the human state. Because it is preferable for the robot to take optimal actions for the interacted human, the robot action policy should also be optimized in reference to human state estimation (and own state). In that instance, it is
important from the viewpoint of control theory [23], [24] that the human state satisfies Markov property in order for the robot to obtain the optimal policy. To incorporate this prior knowledge into our framework, the human state transition in response to the robot action (i.e., dynamics) can be explicitly predesigned as a module and trained like the other modules. Additionally, considering the possibility that this human state may not be intuitively understandable, we have also prepared a module to decode it back to the observation dimension.

Such a conceptual framework is illustrated in Fig. 1. Since this article focuses on the analysis of pHRI, the proposed framework in this article is built only for pHRI from robot to human, although the opposite direction is also important for pHRI optimization. Furthermore, for simplicity, the robot action policy is optimized via behavioral cloning, which is the simplest methodology for optimizing policy [23] (see Section II-D).

Hence, to satisfy the abovementioned requirements, we consider the problem to learn the four modules for pHRI from robot to human. Since optimizing the four modules independently would complicate and interfere with their learning process with the others, this study avoids such an approach and formulates their learning problem as simultaneous learning. Thus, in this section, we derive the learning framework’s foundation, namely a graphical model and its loss function to be minimized.

### B. Simplified VRNN

To extract the latent space $z$ from the high-dimensional observation space $o$, VAE [14], [15] is frequently employed. VAE derives a maximization problem of variational lower bound, and its variant, named $\beta$-VAE, regards it as a constrained optimization problem, which introduces the hyperparameter $\beta$ for regularization to prior information. By integrating VAE with RNNs, such as LSTM [18] and stochastic continuous-time RNN (S-CTRNN) [19], to handle time-series data (e.g., sampled from pHRI), VRNN, so-called VRNN, has been developed [25].

Let us briefly introduce the optimization problem of a simplified VRNN, which excludes several computational graphs to reduce the computational cost. At first, a sequence of observation up to $t$ time step, $o_{<t}$, is compressed to a historical feature $h_t$ by RNNs (originally, the standard RNN is employed) [25]

$$o_{<t} \approx h_t = \text{RNN}(z_t, h_{t-1}), \quad h_0 = 0. \quad (1)$$

To simplify back propagation across the computational graph and distinguish between the latent state estimator and the latent dynamics (see in the following section), this formula ignores an argument $o_t$, which was originally given as $\text{RNN}(o_t, z_t, h_{t-1})$.

With $h_t$, we consider the problem of maximizing the prediction probability of $o_{t+1}$, i.e., $p(o_{t+1} | h_t)$. Assuming that $o_{t+1}$ is generated according to a stochastic latent variable $z_{t+1}$ that is conditional on $h_t$, the variational latent lower bound is given as follows:

$$\ln p(o_{t+1} | h_t) = \int p(o_{t+1} | z_{t+1})p(z_{t+1} | h_t)dz_{t+1}$$

$$= \ln \int q(z_{t+1} | o_{t+1}, h_t)p(o_{t+1} | z_{t+1})$$

$$\geq \mathbb{E}_q(z_{t+1} | o_{t+1}, h_t)[\ln p(o_{t+1} | z_{t+1})]$$

$$- \text{KL}(q(z_{t+1} | o_{t+1}, h_t) \| p(z_{t+1} | h_t))$$

$$= -\mathcal{L}_{\text{vrnn}}$$

(2)

where $p(o_{t+1} | z_{t+1})$, $p(z_{t+1} | h_t)$, and $q(z_{t+1} | o_{t+1}, h_t)$ represent the decoder, time-dependent prior, and encoder, respectively. Note that as well as (1), the decoder is simplified by ignoring the direct dependency of $h_t$, which can be originally given as $p(o_{t+1} | z_{t+1}, h_t)$, for distinguishing the latent state estimator and the latent dynamics. With this structure, $h_t$ would be optimized primarily to improve the representation capability of the state estimator, while the dynamics (and decoder) would be only responsible for taking the latent state one step further. The Monte Carlo method is used to estimate the expectation operation in the first term (usually, with one sample). The second term, Kullback–Leibler (KL) divergence, is frequently computed analytically using a simple stochastic model with its closed-form solution for $p(o_{t+1} | h_t)$ and $q(o_{t+1} | o_{t+1}, h_t)$.

Specifically, $p(o_{t+1} | z_{t+1})$ and $q(z_{t+1} | o_{t+1}, h_t)$ are approximated by deep neural networks (DNNs) with the simple stochastic model like normal distribution. When a new observation $o_{t+1}$ is obtained, $o_{t+1}$ and $h_t$ are fed into the networks and $\mathcal{L}_{\text{vrnn}}$ is computed. $o_{t+1}$ is also fed into RNNs to update $h_t$ to $h_{t+1}$ before a new observation arrives. The total networks including RNNs are updated by minimization of $\mathcal{L}_{\text{vrnn}}$ using a combination of backpropagation following the computational graph for computing $\mathcal{L}_{\text{vrnn}}$ and one of the stochastic gradient descent methods like [35].

### C. Extension for pHRI

On pHRI, the observed information for humans, $o$, is updated based on not just on the history of $o$ but also the robot’s action, $a$, via dynamics between the human and the robot. Furthermore, the robot’s action is generated (stochastically) by its feedback controller based on the current human state and/or feedforward controller based on the history of $a$. Under such a natural problem statement, we extend VRNN to the one with the dynamics and the robot’s policy, which is suitable for representing pHRI.

Specifically, the latent space is explicitly considered as the human state space $s$, i.e., $z = s$. In terms of the dynamics of $s$, we assume the Markovian dynamics described as follows, which is useful for planning the optimal action [36]

$$s_{t+1} = f(s_t, a_t)$$

where $f(\cdot)$ denotes the general nonlinear dynamics, which can be approximated by DNNs. With this, the robot’s policy $\pi$ can be given as the (stochastic) function of $s_{t}$ (i.e., the feedback controller) ideally. However, in practice, it is not always possible to extract $s_t$ that fully satisfies Markov property, therefore, the feedforward component is also needed for more robust control.

To make the feedforward controller, the history of the robot actions, $h^a$ for $a_{<t}$, should be recorded. The history of the human states is distinguished here from $h^o$ by defining $h^s$ for $s_{<t}$.
and cannot be distinguished due to learning.\

\( a | o \) and \( o | (7) \) can be regarded as the minimization denoting the true robot policy. Since \( (8) \) is \( \pi \) and \( h | a \) are therefore introduced for the three KL divergences. \( a | f \), \( RNN - o \), \( \beta \) are explicitly distinguished in the formu-

In Fig. 2. The current state satisfying Markov property \( s \) is estimated from the current observation \( o \) and the past state sequence \( h_{t-1}^s \), and the robot action \( a \), which depends on \( s \) (and the past action sequence \( h_{t-1}^a \)), acts to the dynamics \( f \) to update the state at the next time \( s_{t+1} \). Afterwards, \( s_{t+1} \) is decoded to the next observation \( o_{t+1} \). In addition, \( h_{t+1}^s \) and \( h_{t+1}^a \) are also updated according to (4) and (5), respectively. As a remark, the encoder for \( s \) can be regarded as the human state estimator, \( q(s_t | o_t, h_{t-1}^s) \).

From this graphical model, we can find that \( a \) can be embedded into (2) as a part of \( z \). Specifically, (2) is rederived as follows:

\[
\begin{align*}
\ln p(a_{t+1} | h_{t-1}^a, h_{t-1}^s) \\
= \ln \int \int p(a_{t+1} | s_{t+1}) \\
\times p(s_t | h_{t-1}^s)p(a_t | h_{t-1}^a)ds_t da_t \\
= \ln \int \int q(s_t | o_t, h_{t-1}^s)\pi(a_t | s_t, h_{t-1}^a) \\
\times p(o_{t+1} | s_{t+1}) \\
\times \frac{p(s_t | h_{t-1}^s)p(a_t | h_{t-1}^a)}{q(s_t | o_t, h_{t-1}^s)\pi(a_t | s_t, h_{t-1}^a)}ds_t da_t \\
\geq \mathbb{E}_{q(s_t | o_t, h_{t-1}^s)\pi(a_t | s_t, h_{t-1}^a)}[\ln p(o_{t+1} | s_{t+1})]
\end{align*}
\]

where \( p(a_{t+1} | s_{t+1}), p(s_t | h_{t-1}^s), \) and \( p(a_t | h_{t-1}^a) \) denote the decoder and the time-dependent priors for state and action, respectively. Note that the variables given as conditions are one time earlier than those in (2) because they are explicitly advanced by one step due to the dynamics \( f \).

Several studies have been proposed to construct latent dynamics [16], [17], but by considering parts of the latent variables in VAE (or VRNN in our case) as the robot action, as in this study, a (time-dependent) prior for the robot action policy, \( p(a_t | h_{t-1}^a) \), is introduced. This prior can be utilized not only for regularization of policy optimization but also as a feedforward controller to achieve robust control even in the presence of sensing failures [37].

D. Auxiliary Optimization for Robot Action Space

Although \( s \) and \( a \) are explicitly distinguished in the formulation of (6), when the total system is approximated by DNNs (and RNNs), \( s \) and \( a \) cannot be distinguished due to learning with backpropagation to the whole networks. Therefore, an optimal control problem for \( a \) (and \( \pi \)), such as reinforcement learning [24], should be provided as an auxiliary optimization to clarify the role of \( a \).

To this end, the simplest behavioral cloning is used in this article for simplicity. The following optimization, in particular, is also considered:

\[
\begin{align*}
\mathcal{L}_{\text{act}} &= \text{KL}(\pi^*(a_t) || \pi(a_t | s_t, h_{t-1}^a)) \\
&\quad \times \mathbb{E}_{a_t \sim \pi}[\ln \pi(a_t | s_t, h_{t-1}^a)]
\end{align*}
\]

where \( \pi^* \) denotes the true robot policy. Since \( \pi^* \) is at least a black-box function that can sample \( a \), Monte Carlo approximation is available even if its KL divergence cannot be computed analytically. Furthermore, by removing the term unrelated to the optimization of \( \pi \), we actually obtain the expected value (i.e., sample mean) of the negative log-likelihood of \( \pi \).

Following \( \beta \)-VAE [15], the minimization problem for \( \mathcal{L}_{\text{dyn}} \) in (6) and \( \mathcal{L}_{\text{act}} \) in (7) can be regarded as the minimization problem constrained with the three KL divergences. The weights \( \beta_{1,2,3} \) are therefore introduced for the three KL divergences. Consequently, given the trajectories of the tuples of the current observation, the robot’s action, and the updated observation, \( (o_t, a_t, o_{t+1}) \), the loss function to be minimized is given as follows:

\[
\begin{align*}
\mathcal{L} &= -\ln p(a_{t+1} | s_{t+1} = f(s_t, \tilde{a}_t)) \\
&\quad + \beta_1 \text{KL}(q(s_t | o_t, h_{t-1}^s) || p(s_t | h_{t-1}^s)) \\
&\quad + \beta_2 \text{KL}(\pi(a_t | s_t, h_{t-1}^s) || p(a_t | h_{t-1}^a)) \\
&\quad - \beta_3 \ln \pi(a_t | s_t, h_{t-1}^a) \\
\text{s.t.} \\
&\quad s_t \sim q(s_t | o_t, h_{t-1}^s), \tilde{a}_t \sim \pi(a_t | s_t, h_{t-1}^a).
\end{align*}
\]
Note that \( h_{t-1}^x \) and \( h_{t-1}^y \) are updated using RNNs, as described in (4) and (5), at each time step. Additionally, \( \tilde{a}_t \) is defined to distinguish it from \( a_t \sim \pi^* \).

Because we have the ideal \( a_t \) in this setting, we can input it to the dynamics (and decoder). However, to solve the optimal control problems, the error backpropagation of the ideal observation transition can be used [38]. In order to make this approach conceivable, our implementation uses \( \tilde{a}_t \) as input to the dynamics. However, we have to notice that, to learn the dynamics correctly, \( \tilde{a}_t \) should be the same as or similar to the action used for interaction with the real environment. Because of supervised learning, the robot action policy can be easily trained using behavioral cloning, thereby satisfying this assumption.

III. DESIGN OF RNN SUITABLE PERIODIC pHRI

A. Motivation

As described earlier, the proposed framework requires RNNs for \( h_{t}^x \) and \( h_{t}^y \). The first bottleneck of the proposed framework is that, although the graphical model is simplified from the original [25], a huge complex computational graph is still constructed when BPTT [39] is applied for optimizing RNNs. This increases the computational cost and introduces learning instability, particularly in problems where RNNs cannot be directly optimized. Although it is possible to keep the computational cost by using truncated BPTT [26], it has been reported that the truncation causes a bias in the learning results [27].

We use RC [28], [29] as one of the RNNs to simplify the backpropagation and make the training of the system easier and more stable (see Fig. 3). Although RC can provide sufficient time representation due to numerous random neurons, unlike LSTM [18], it is difficult to handle the prediction problems with long time delays. For periodic pHRI, RC needs to be extended to capture the information of the periodic past. We examine the extension of RC into a complex domain in this study. The decay of amplitude and the rotation of phase represent the behavior in the complex domain, and the rotation of phase, in particular, leads to the utilization of periodic past information.

B. Complex-Valued RC

RC is a model built with fixed random weights, \( w_{in} \) and \( w_{rc} \), and biases, \( b \), and updates its internal state \( h \) according to input \( u \) as follows:

\[
h_{t+1} = (1 - \gamma) \circ h_t + \gamma \odot \tanh(w_{in}^\top u_t + w_{rc}^\top h_t + b)
\]

where \( \gamma \in (0, 1] \) represents the leaky integrator and \( \odot \) denotes the element-wise multiplication. For numerical stability, the spectral radius of \( w_{rc} \) should be less than one.

If RC has sufficient neurons and \( w_{in}, w_{rc}, b, \) and \( \gamma \) are randomly assigned, a portion of the neurons can adequately capture the features of the time-series data. However, in principle, the RC defined earlier has an exponential decay of \( h \), making long-term characteristics difficult to represent (e.g., periodicity). To address this drawback and make RC suitable for periodic pHRI, we focus on complex-valued neural networks [32], [33], which exhibit oscillatory behavior between real and imaginary values.

When RC is augmented into a complex domain, such oscillatory behavior allows for the representation of periodic pHRI.

Specifically, \( w_{in}, w_{rc}, \) and \( b \) can be given randomly as well even in the complex domain. Since the spectral radius encompasses the complex matrix, numerical stability can be easily guaranteed [40] even for the complex domain. As for the activation function, \( \tanh \), the phase-amplitude version [41] is employed

\[
\tanh(x) = \tanh(|x|) \exp(i\phi(x))
\]

where \( \phi(x) \) denotes the phase of \( x \) in complex domain.

In contrast, \( \gamma \) should be carefully designed since \( 1 - \gamma \) may violate \( |1 - \gamma| < 1 \) for numerical stability (see Fig. 4). To avoid this violation, the amplitude of \( \gamma, |\gamma| \), is first randomly given. Afterwards, the upper bound of the phase of \( \gamma, \bar{\phi}(\gamma) \), is derived as follows:

\[
(1 - |\gamma| \cos \bar{\phi}(\gamma))^2 + (|\gamma| \sin \bar{\phi}(\gamma))^2 = 1
\]

\[
\therefore \bar{\phi}(\gamma) = \cos^{-1} \frac{|\gamma|}{2}.
\]

The phase of \( \gamma, \phi(\gamma) \), is, therefore, randomly given within \([0, \bar{\phi}(\gamma)]\).

C. Verification of Qualitative Behavior

With the abovementioned implementation, we can see the oscillatory behavior in Fig. 5. The free-response after 100 steps of adding random inputs shows that the internal states continue to oscillate without decay. Indeed, by analyzing the frequency at this time through fast Fourier transform (FFT), multiple natural
The robot responds to the following three types of commands: motor angle, angular velocity, and torque. These are generated by feedforward rotational motion and admittance (i.e., feedback) control (for details, see Appendix A).

A human partner holds another end of the rope at a distance of around 1.5 m from the robot. To increase the variation of the motions to be analyzed, the partner generates the motions by switching between using the left or right arm. Note that when the rope was too long, the movement’s uncertainty was likely to be high. Thus, the rope length was adjusted so that it does not touch the ground when grasped.

For human observation, we use a ZED2 stereo camera from Stereolab. It can detect the 3-D coordinates of the 18 human feature points, as well as the center position and velocity of humans; 60 dimensions are acquired as the human observation space in total. Since these 60-dimensional features can be measured with 30 fps while the ODrive board accepts the commands with over 100 fps, the system frequency is set to be 30 fps (i.e., the time step \( \Delta t = 1/30 \) s) according to the lower one. Note that there may be other observations that more directly show the human state (e.g., the angle of robot), but we only use this observation in order to evaluate the competence of extracting the latent state space while keeping the system configuration as basic as feasible.

Each time-series trajectory has 900-time steps (i.e., around 30 s). In each trajectory, the robot operates in four different conditions, i.e., rotation or swinging with slow or fast speed (see Appendix B), and the partner follows the robot motion accordingly. Additionally, by following the instruction from the robot to switch the arm that mainly moves, there would be eight conditions in total. Since all the conditions are determined by the robot, the label of each trajectory is known. All the motion patterns are periodic, and therefore, CRC in the proposed framework is expected to properly represent the periodic time-series data.

We will verify whether the proposed framework can classify these conditions in the revealed latent state space. On each condition, eleven trajectories are recorded for training (for a total of 88 trajectories with 9900 data for all conditions), as well as three trajectories for validation and three trajectories for testing. The classification problem is not solved in this experiment, that is, the known labels are not explicitly used for learning. Instead, they are used to distinguish clusters in the analysis.

For the analysis, the proposed framework compresses the 60-dimensional observation into three-dimensional latent state space (for easy visual analysis). Note that the nonlinear mapping from the observation to the latent state is obtained through unsupervised learning, the meaning of each component in the state space is ambiguous. Despite the fact that the topology of the observation data is preserved even in the state space, we can focus our analysis on the topology of the clusters that would be built for the eight conditions.

IV. EXPERIMENT

A. Rope Rotation/Swinging

As an example of periodic pHRI, in this article, we analyze a rope-rotation/swinging task. The system configuration for this task is shown in Fig. 6. In addition, the system-dependent variables in the proposed framework, \( o \) and \( a \) (\( s \) additionally), are specified in Table I.

One side of the rope is attached to a one-joint robot with 0.5 m arm length and a brushless dc motor (D5065) controlled via ODrive board both of which are developed by ODrive Robotics. Thanks to the absence of gears, this motor has back-drivability.
Fig. 5. Example of oscillatory behavior on complex-valued RC: the bias term $b$ is set to zero in order to unify the convergence point of $h$ to zero for easier viewing; (a) although the standard RC excited only when the random inputs were applied until 100 steps, the complex-valued RC achieved the periodic oscillation without divergence during the free-response; indeed, FFT analyzes in (b) and (c) complex-valued RC has the natural frequencies for the respective neurons, and amplifies $h$ with multiple frequencies. (a) Trajectories for all the neurons. (b) FFT analysis with random inputs. (c) FFT analysis of free response.

Fig. 6. Experimental environment: a one-joint robot with a brushless DC motor attempts to rotate or swing a rope according to the specified conditions; a human partner attempts to follow its movements; the human motion can be detected by a ZED2 stereo camera.

Fig. 7. Overview of information processing in the proposed framework: this framework takes the current observation $o_t$ as inputs and produces the exerted action $a_t$ and the estimated state $s_t$ as output; a human state estimation module returns the estimated state $s_t$ based on the current observation $o_t$ and the previous states $h_{t-1}^q$; a robot policy module infers the exerted action $a_t$ in the current (estimated) state $s_t$ and the past actions $h_{t-1}^a$; a dynamics module updates the state $s_t$ to the next one $s_{t+1}$ by acting the outputted action $a_t$, then resulting in prediction of the new observation $o_{t+1}$ through a decoder; except for RNN modules, we use FCNs to map inputs to outputs; Furthermore, we apply nonlinear transformations to output parameters in the correct domain (e.g., a scale parameter in positive domain); note that the FCNs with two input arrows concatenate them as a vector signal.

Table II

| Symbol | Meaning | Value |
|--------|---------|-------|
| $\alpha$ | Learning rate | 0.0001 |
| $\beta_1$ | Weight for state regularization | 0.1 |
| $\beta_2$ | Weight for policy regularization | 0.1 |
| $\beta_3$ | Weight for auxiliary policy optimization | 1.0 |

For initializing CRCs in the proposed framework, each module has $N_{rc} = 1000$ neurons. All the parameters are sampled from uniform distribution $\in [-1, 1]$. Furthermore, the parameters representing the recurrent network dynamics, $w_{in}, w_{rc}$, and $b$, have randomly sparse connections: each component of them is forced to be zero with probability $1 - 1/N_{rc}$ based on [45].

Afterward, we transform them so that they are all within the ranges that must be satisfied for each variable. That is, $w_{rc}$ should make its spectral radius less than 1 ($0.9$ in this article); $w_{in}$ and $b$ are divided by $1 + \text{rows}(w_{in})$ (rows denotes the number of rows) to normalize them; $|\gamma|$ should be set to (0, 1]; and $\phi(\gamma)$ should be set to $[0, \phi(\gamma)]$ as obtained in (12). Note that if the history information outputted from (C)RC is directly concatenated with $o$ or $s$, the history may become dominant due to its large dimensionality. The output from (C)RC is, therefore, fed into FCNs before concatenation to modify the number of dimensions and scale.

For training the networks, we used td-AmsGrad, one of the state-of-the-art optimizers that are robust to noise and outliers [46], [47], with default parameters except learning rate $\alpha$. We notice again that the parameters of (C)RC (i.e., $w_{in}, w_{rc}, b$, and $\gamma$) are not updated.

Table II summarizes hyperparameters for learning. To stabilize unsupervised learning, the learning rate $\alpha$ was set lower than layers with 100 neurons for each, and the combination of layer normalization [43] and rectified linear unit (ReLU) function as a nonlinear activation function. The decoder $p(o \mid s)$ is parameterized by diagonal student-t distribution for robust learning against observation noise and fluctuations in human behavior [44], and the other distributions are parameterized by a diagonal normal distribution.
The learning curve of the validation data generally converged. The batch size was limited to half of the number of trajectories of the training data due to memory limitation. \( \beta_{1,2} \) were adjusted to less than 1 to prioritize the prediction accuracy. \( \beta_3 \) was fixed at 1 to achieve the same level of priority for the auxiliary optimization, although we introduced \( \beta_3 \) for generality.

C. Learning Performance

The proposed framework consists of VRNN and three additional components: “D” explicit human–robot dynamics, “A” auxiliary policy optimization, and “C” CRC for periodicity. Note that, in the case with \(-D+A\), \( \tilde{a}_t \) is replaced to one vector with the same dimension to forcibly exclude the effects of \( \tilde{a}_t \) on \( s_{t+1} \). We investigate their effects in terms of the learning performance, specifically mean squared error (MSE) between true and predicted human observations and/or robot actions. The neural networks shown in Fig. 7 are trained after being initialized with 20 random seeds for each of eight conditions (i.e., with and without the three components). The performances for the validation data in the learning process are depicted in Fig. 8. Additionally, the performances for the test data are evaluated after training and are summarized in Table III.

As can be seen in Fig. 8(a), the framework failed to learn the robot policy unless “D” and “A” are activated. As mentioned earlier, “D” alone did not work at all because the robot action \( a \) is mixed into the human state \( s \) without explicit discrimination of them. On the other hand, if “A” is activated, we expect that learning can be accomplished in a supervised learning manner. However, it failed probably because the next robot action cannot be predicted based on the robot’s previous action sequences. By “A” clarifying the roles of \( a \) and \( s \), and by “D” conveying the gradient information about the prediction of the human observation to the robot policy, the proposed framework makes the robot policy learnable. The gradient information is also traced back to the \( s \) fed into the robot policy, implying the benefits of \( s \) reshaped to the one suitable for \( a \) and the dynamics.

From Fig. 8(b), we can say that the proposed CRC (i.e., ones with +C labels) outperformed the standard RC (i.e., ones with −C labels). Although “C” caused larger MSE in the early stages of learning that was reversed around the 25 epoch. Afterwards, the standard RC mostly remained on the local solution at that time; in contrast, the proposed CRC continued to reduce MSE even at 100 epoch (i.e., the end of learning). The learning delay was probably caused by the more complex dynamics that were difficult untangle. The final result was due to the fact that, as expected, parts of the complex dynamics were better suited to represent the periodic motions of this rope-rotation/swinging task.
D. Latent Space Analysis

From the preceding results, it was confirmed that the introduction of CRC enhanced prediction accuracy. The ability to represent periodic time-series data, which considerably contributes to the building of latent state space, is closely related to CRC. We visualize the latent state space in the proposed framework with all components for a more in-depth analysis of the utility of CRC. Since it is difficult to include all the data due to the limitation of space, the three trajectories for each condition in the test data are converted to $s$ using the model initialized with the second random seed, which resulted in a similar accuracy to the mean and median in Table III. The transformed trajectories in the latent state space are illustrated in Fig. 9. Note that the $-D-A-C$ model initialized with the second random seed, the accuracy of which was also close to the statistical metrics in Table III, was compared to the conventional method. Furthermore, because the eight clusters usually overlapped from any angle, analyzing them in a 3-D plot was difficult. Thus, for the sake of clarity, all trajectories were divided into two groups for each condition (e.g., swinging or rotation).

Despite the role and scale of each axis differed in the two methods, the swinging and rotating trajectories were almost orthogonally ordered as a result of unsupervised learning (even for each random seed). While these two motions should have identical sceneries if their moments stripped out, it is believed that such an orthogonal state space was obtained by appropriately considering the time series.

The proposed framework symmetrically separated the trajectories driven primarily by the left and right arms. However, with the conventional framework, some of the trajectories that appear to be driven mainly by the left arm seem to be mixed up in the cluster for the right arm. To confirm this, only the $(S, L, S)$ and $(S, R, S)$ trajectories were extracted and drawn on Fig. 10. Indeed, the two clusters were clearly separated by the proposed framework, but in the conventional framework, one of the trajectories of $(S, L, S)$ was misclassified into the cluster on $(S, R, S)$. This result suggests that the difference in the ability to represent the time-series data led to such correct and incorrect clustering and also affected the prediction accuracy.

Finally, we focus on the two groups split according to the speed (i.e., slow or fast). Compared to the other conditions, these two were not distinguished well even with both frameworks,
but the cluster for the fast motions was in the cluster for the slow motions. To clarify that, the first attached video shows the motions corresponding to the test data and the animations of the trajectories. From the video, we found that, when rotating the rope, the cluster for the fast motions was certainly in the cluster for the slow motions. Furthermore, when swinging the rope, the cluster for the slow motions was slightly shifted from the one for the fast motions while significantly reducing its amplitude. The inclusion relationship between the rotation motion clusters can be attributed to the fact that their basic motions were the same, but in the fast motion, the motion was compacted by actively using the rope’s inertia.

In summary, although the conventional framework was able to extract the characteristics of each cluster to some extent, the proposed framework succeeded in clustering with higher accuracy, as shown in Fig. 10. Since it can be easily assumed that misclassification affects the prediction accuracy of observation, it is suggested that the CRC, which greatly contributed to the improvement (see Fig. 8 and Table III), was able to properly capture the characteristics of periodic time-series data.

E. Discussion

1) Parameter Dependencies: In this experiment, the parameters and the network architecture were determined empirically based on past studies and experiences. Although the comparison was done in a more qualitative fashion, the impact of these factors on the performance (e.g., optimal values and robustness) is anticipated to be investigated. For example, it has been empirically proven that $\beta_{1,2}$ should be less than 1 to prioritize the prediction accuracy, although the effective value range is unclear.

The design theory of CRC, which is the most essential component of our proposal, needs to be developed. The natural frequencies of the CRC, as illustrated in Fig. 5 should be significantly dependent on the design of the network. The task dynamics can be represented more easily if task-specific frequencies are completely included in the CRC’s frequencies; otherwise, we will fail in/suffer from transforming the CRC dynamics to the task dynamics. Although CRC keeps more natural frequencies than the regular RC, developing task-specific structures or generalizing them to include more natural frequencies to represent arbitrary tasks would be desirable. As a result, future study will focus on refining the standard design theory for RC (e.g., [30], [48]) and/or meta-optimization [49], [50] to that for CRC with explicit consideration of the complex numbers.

2) Refinement of Optimal Control Problem: Behavioral cloning was solved concurrently in the proposed framework to aid in the optimization of the robot policy. As a result, while the accuracy of action estimation for the test data was sufficiently high (see Table III), the actual control performance has certain shortcomings. Specifically, behavioral cloning suffers from an unresolved problem known as covariate shift and compounding error, in which the state distribution gradually shifts to an inexperienced space during the demonstration with the trained policy and fails to generate optimal actions (also see Appendix C).

It is desirable to integrate a more sophisticated optimal control problem as an auxiliary optimization to address this issue. For example, DART [51], a behavioral cloning extension, might be incorporated to obtain a more robust policy against noise cause by human diverse behaviors. Additionally, model-based reinforcement learning with the learned dynamics and model predictive control can be employed to plan the optimal action sequences, which reproduce the ideal pHRI (e.g., to smoothly follow the human movements without interference), through error backpropagation [36]. The planned action sequences will eventually be transferred into the robot policy in our framework [52]. In any case, developing advanced robot controllers for pHRI based on the proposed framework with high (periodic) expressive capabilities obtained in this article is required in the future.

V. Conclusion

This article developed a new data-driven framework for analyzing periodic pHRI in latent state space using the modified VRNN derivation and CRC. We specifically modified VRNN to explicitly include the latent state dynamics updated based on the robot’s action, which were also inferred from data, and to identify it as a separate module from the human state estimator. As the RNN model suitable for representing periodic motions, we augmented the standard RC in the real domain to the one in the complex domain, named CRC. The phase rotation in complex domains intuitively enables CRC to represent the periodic time-series data. A rope-rotation/swinging experiment was used to validate the proposed framework. As a result, the proposed framework, which comprised all three important components, attained the maximum accuracy in observation and action prediction. Furthermore, the proposed framework achieved the well-distinguished latent state space for analyzing the eight-type periodic motions, although the conventional method confused one of them.

Because the primary focus of this article was on human motion analysis and classification, the robot policy was predesigned. However, in real-world applications, the robot policy may be desired to be optimized toward the desired human state. To this end, future studies will necessitate the inference of the human internal modules in order to predict overall pHRI. We anticipate
that by obtaining them, the robot will be able to optimize its policy through model-based reinforcement learning.

**APPENDIX**

### A. Motor Control

The dynamics defined by the virtual impedance is employed to control the motor. For the following dynamics, an inertia $m$, a damping coefficient $c$, and a spring coefficient $k$ are defined:

$$m\ddot{\theta} + c(\dot{\theta} - \dot{\theta}_{\text{ref}}) + k(\theta - \theta_{\text{ref}}) = \tau$$

where the actual angle $\theta$ and the angular velocity $\dot{\theta}$ can be measured using an encoder, and the external torque $\tau$ is estimated as follows. $\theta_{\text{ref}}$ and $\dot{\theta}_{\text{ref}}$ are given to track the reference trajectory (see Appendix B).

Because we use the brushless dc motor to control the robot directly, the measured current $i$ is proportional to the torque applied. However, because the controller’s internal torque cannot be separated from it, the system simply assumes that the increase or decrease from the time average of the exerted torque $\bar{\tau}$ corresponds to $\tau$

$$\tau_i = \kappa i$$

$$\bar{\tau} \leftarrow \eta \bar{\tau} + (1 - \eta)\tau_i$$

$$\tau = \tau_i - \bar{\tau}$$

where $\kappa$ denotes the hardware torque constant and $\eta$ is designed for exponential moving average.

By Euler integration of the derived acceleration $\ddot{\theta}$, the command angular velocity and angle are obtained as follows:

$$\dot{\theta}_{\text{cmd}} = \dot{\theta} + \dot{\theta}\Delta t$$

$$\theta_{\text{cmd}} = \theta + \dot{\theta}_{\text{cmd}}\Delta t.$$  

In order to weaken the resistance to $\tau$, the following torque command is heuristically given to the controller

$$\tau_{\text{cmd}} = -\mu \bar{\tau}$$

where $\mu$ denotes the gain. Note that since these command values differ in scale and are unsuitable for learning, the robot’s action space is defined as the disparities between them from the preceding command values.

Table IV lists the parameters associated with the abovementioned controller design. These were hand-tuned before the experiment to provide natural interaction with a human and to maintain the ability to follow the motion patterns.

### B. Motion Generation

We design two simple velocity profiles to generate the reference trajectory for the rope rotation/swinging, as shown below. By Euler integration of the designed velocity $\dot{\theta}_{\text{ref}}$, the reference angle $\theta_{\text{ref}}$ is derived as follows:

$$\theta_{\text{ref}} \leftarrow \theta_{\text{ref}} + \dot{\theta}_{\text{ref}}\Delta t.$$  

Note that the motor we used counts the angle throughout a single rotation without resetting.

A trapezoidal velocity profile is used for the rope rotation. Given an acceleration time $t_a$, a constant velocity time $t_c$, and a maximum velocity $v$, $\dot{\theta}_{\text{ref}}$ at the current time $t$ is given

$$\dot{\theta}_{\text{ref}} = \begin{cases} v \frac{t}{t_a} & 0 \leq t < t_a \\ v & t_a \leq t \leq t_a + t_c \\ v - v \frac{t - (t_a + t_c)}{t_a} & t_a + t_c < t < 3t_a + t_c \\ -v & 3t_a + t_c \leq t \leq 3t_a + 2t_c \\ -v + v \frac{t - (3t_a + 2t_c)}{t_a} & 3t_a + 2t_c < t \leq 4t_a + 2t_c \end{cases}.$$  

This profile is repeated by resetting $t$ until the trial ends.

For the rope swinging, a cosine velocity profile is employed. Given a swing period $t_s$ and a swing amplitude of (i.e., maximum velocity $v$), the velocity profile is given as follows:

$$\dot{\theta}_{\text{ref}} = v \cos \left(2\pi \frac{t}{t_s}\right).$$

The parameters associated with the abovementioned velocity profiles are listed in Table V. These were hand-tuned before the experiment to enable a human to follow the generated motions.

### C. Demonstration With Learned Policy

We demonstrate the proposed method (i.e., the +D+A+C model used to draw Fig. 9) with the learned policy instead of the abovementioned controller. This should be noted that the abovementioned controller took the lead in determining the motion pattern with the human, but the learned policy needs to classify the motion pattern while monitoring and following the human motion. This means that the domain for the demonstration differs from the training data, and especially in behavioral cloning, which is sensitive to such changes, compounding error generated incorrect control signals that interfered with the human motions. To alleviate this problem, we improved the back-drivability by reducing the PID gains of the motor and using the average of the observation and the action taken by the learned policy as the command value, instead of directly passing the action to the
Fig. 11. Trajectories of demonstrations: the top three show the error between the implemented controller and the learned policy for each label; the bottom shows the robot angle during the demonstration with auxiliary lines on (a) ±π/2 as range of motion and (b) each rotation; the desired periodic motions were obtained in both situations; with the exception of the angular velocity, the learned policy outputted similar values to the implemented controller. (a) (S, L, S). (b) (R, R, S).

motor. This should eliminate the compounding error following human feedback, as long as it does not interfere with human motions.

The demonstration results assuming (S, L, S) and (R, R, S) motions are shown in Fig. 11 and the second attached video. Although the whole motion pattern was jerky than that of the implemented controller, we found that the error of the angle command, in particular, was sufficiently suppressed in both cases. In other words, with no compounding error, the policy learned using the proposed method can imitate the controller with a certain degree of accuracy. However, the error of the angular velocity command was often large. This may be due to the noisy supervisory signal by the implemented controller, which deteriorated the learning accuracy. Further improvement of the latent space is projected to result in a more qualitative understanding of pHRI, yielding more robust policy and dynamics to be learned.

REFERENCES

[1] P. Beckerle et al., “A human-robot interaction perspective on assistive and rehabilitation robotics,” Front. Neurorobot., vol. 11, pp. 1–24, 2017.
[2] T. Inamura and Y. Mizuochi, “Sigverse: A cloud-based VR platform for research on multimodal human-robot interaction,” Front. Robot. AI, vol. 8, pp. 1–19, 2021.
[3] P. Higgins et al., “Towards making virtual human-robot interaction a reality,” in Proc. Int. Workshop Virtual, Augmented, Mixed-Reality Hum. Robot Interact., 2021.
[4] C. D. Kidd and C. Breazeal, “Robots at home: Understanding long-term human-robot interaction,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2008, pp. 3230–3235.
[5] O. Liu, D. Rakita, B. Mutlu, and M. Gleichert, “Understanding human-robot interaction in virtual reality,” in Proc. IEEE Int. Symp. Robot Hum. Interactive Commun., 2017, pp. 751–757.
[6] M. Khoramshahi and A. Billard, “A dynamical system approach to task-adaptation in physical human–robot interaction,” Autom. Robots, vol. 43, no. 4, pp. 927–946, 2019.
[7] F. Ferraguti, C. Talignani, L. Sabattini, M. Bonfè, C. Fantuzzi, and C. Secchi, “A variable admittance control strategy for stable physical human-robot interaction,” Int. J. Robot. Res., vol. 38, no. 6, pp. 747–765, 2019.
[8] S. Itadera, T. Kobayashi, J. Nakanishi, T. Aoyama, and Y. Hasegawa, “Towards physical interaction-based sequential mobility assistance using latent generative model of movement state,” Adv. Robot., vol. 35, no. 1, pp. 64–79, 2021.
[9] X. Yu et al., “Bayesian estimation of human impedance and motion intention for human-robot collaboration,” IEEE Trans. Cybern., vol. 51, no. 4, pp. 1822–1834, Apr. 2021.
[10] W. He, C. Xue, X. Yu, Z. Li, and C. Yang, “Admittance-based controller design for physical human–robot interaction in the constrained task space,” IEEE Trans. Auton. Sci. Eng., vol. 17, no. 4, pp. 1937–1949, Oct. 2020.
[11] T. Kobayashi, E. Dean-Leon, J. R. Guadarrama-Olvera, F. Bergner, and G. Cheng, “Whole-body multicontact haptic human–humanoid interaction based on leader–follower switching: A robot dance of the ‘box step’,” Adv. Intell. Syst., vol. 4, no. 2, 2021, Art. no. 2100038.
[12] Q. Leboutet, E. Dean-Leon, F. Bergner, and G. Cheng, “Tactile-based whole-body compliance with force propagation for mobile manipulators,” IEEE Trans. Robot., vol. 35, no. 2, pp. 330–342, Apr. 2019.
[13] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, vol. 521, no. 7553, pp. 436–444, 2015.
[14] D. P. Kingma and M. Welling, “Auto-encoding variational Bayes,” in Proc. Int. Conf. Learn. Representations, 2014.
[15] I. Higgins et al., “β-VAE: Learning basic visual concepts with a constrained variational framework,” in Proc. Int. Conf. Learn. Representations, 2016.
[16] D. Hafner et al., “Learning latent dynamics for planning from pixels,” in Proc. Int. Conf. Mach. Learn., 2019, pp. 2555–2565.
[17] T. Li, R. Calandra, D. Pathak, Y. Tian, F. Meier, and A. Rai, “Planning in learned latent action spaces for generalizable legged locomotion,” IEEE Robot. Autom. Lett., vol. 6, no. 2, pp. 2682–2689, Apr. 2021.
[18] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural Comput., vol. 9, no. 8, pp. 1735–1780, 1997.
[19] S. Murata, J. Namikawa, H. Arie, S. Sugano, and J. Tani, “Learning latent representation in human–robot interaction with explicit consideration of periodic dynamics,” Front. Robot. AI, vol. 5, no. 7553, pp. 436–444, 2015.
[20] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction. Cambridge, MA, USA: MIT Press, 2018.
[25] J. Chung, K. Kastner, L. Dinh, K. Goel, A. C. Courville, and Y. Bengio, “A recurrent latent variable model for sequential data,” *Adv. Neural Inf. Process. Syst.*, vol. 28, pp. 2980–2988, 2015.

[26] G. Puskorius and L. Feldkamp, “Truncated backpropagation through time and kalman filter training for neurocontrol,” in *Proc. IEEE Int. Conf. Neural Netw.*, 1994, vol. 4, pp. 2488–2493.

[27] C. Tallec and Y. Ollivier, “Unbiasing truncated backpropagation through time,” 2017, *arXiv:1705.08209*.

[28] H. Jaeger and H. Haas, “Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication,” *Science*, vol. 304, no. 5667, pp. 78–80, 2004.

[29] M. Lukoszевич and H. Jaeger, “Reservoir computing approaches to recurrent neural network training,” *Comput. Sci. Rev.*, vol. 3, no. 3, pp. 127–149, 2009.

[30] C. Gallicchio, A. Micheli, and L. Pedrelli, “Deep reservoir computing: A critical experimental analysis,” *Neurocomputing*, vol. 268, pp. 87–99, 2017.

[31] T. Kobayashi, “Practical fractional-order neuron dynamics for reservoir computing,” in *Proc. Int. Conf. Artif. Neural Netw.*, 2007, pp. 1061–1066.

[32] S. L. Goh and D. P. Mandic, “An augmented CRTRL for complex-valued recurrent neural networks,” *Neural Netw.*, vol. 20, no. 10, pp. 1550–1560, Oct. 1990.

[33] Y. Xia, D. P. Mandic, M. M. Van Hulle, and J. C. Principe, “A complex echo state network for nonlinear adaptive filtering,” in *Proc. IEEE Workshop Mach. Learn. Signal Process.*, 2008, pp. 404–408.

[34] A. Hirose, “Continuous complex-valued back-propagation learning,” *Electron. Lett.*, vol. 28, no. 20, pp. 1854–1855, 1992.

[35] L. Ba, J. R. Kiros, and G. E. Hinton, “Layer normalization,” 2016, *arXiv:1607.06450*.

[36] H. Takahashi, T. Iwata, Y. Yamanaka, M. Yamada, and S. Yagi, “Student-T variational autoencoder for robust density estimation,” in *Proc. Int. Joint Conf. Artif. Intell.*, 2018, pp. 1324–1327, Mar. 2022.

[37] T. Kobayashi, “Towards deep robot learning with optimizer applicable to non-stationary problems,” in *Proc. IEEE/SICE Int. Symp. Syst. Integration (SII)*, 2021, pp. 190–194.

[38] A. Rodan and P. Tino, “Minimum complexity echo state network,” *IEEE Trans. Neural Netw.*, vol. 22, no. 1, pp. 131–144, Jan. 2010.

[39] K. C. Chatzidimitriou and P. A. Mitkas, “Adaptive reservoir computing through evolution and learning,” *Neurocomputing*, vol. 103, pp. 198–209, 2013.

[40] T. Aotani, T. Kobayashi, and K. Sugimoto, “Meta-optimization of bias-variance trade-off in stochastic model learning,” *IEEE Access*, vol. 9, pp. 148783–148799, 2021.

[41] M. Laskey, J. Lee, R. Fox, A. Dragan, and K. Goldberg, “Dart: Noise injection for robust imitation learning,” in *Proc. Conf. Robot Learn.*, 2017, pp. 143–156.

[42] S. Bansal, R. Calandra, K. Chua, S. Levine, and C. Tomlin, “MBMF: Model-based priors for model-free reinforcement learning,” 2017, *arXiv:1709.03153*.