Mixing Actual and Predicted Sensory States Based on Uncertainty Estimation for Flexible and Robust Robot Behavior

Shingo Murata[1(✉)], Wataru Masuda[1], Saki Tomioka[1], Tetsuya Ogata[2], and Shigeki Sugano[1]

[1] Department of Modern Mechanical Engineering, Waseda University, Tokyo, Japan
murata@sugano.mech.waseda.ac.jp

[2] Department of Intermedia Art and Science, Waseda University, Tokyo, Japan

Abstract. In this paper, we propose a method to dynamically modulate the input state of recurrent neural networks (RNNs) so as to realize flexible and robust robot behavior. We employ the so-called stochastic continuous-time RNN (S-CTRNN), which can learn to predict the mean and variance (or uncertainty) of subsequent sensorimotor information. Our proposed method uses this estimated uncertainty to determine a mixture ratio for combining actual and predicted sensory states of network input. The method is evaluated by conducting a robot learning experiment in which a robot is required to perform a sensory-dependent task and a sensory-independent task. The sensory-dependent task requires the robot to incorporate meaningful sensory information, and the sensory-independent task requires the robot to ignore irrelevant sensory information. Experimental results demonstrate that a robot controlled by our proposed method exhibits flexible and robust behavior, which results from dynamic modulation of the network input on the basis of the estimated uncertainty of actual sensory states.

Keywords: Recurrent neural networks · Uncertainty · Robot · Neurorobotics

1 Introduction

Flexible and robust behavior is crucial for autonomous robots that are expected to work in the same environments as people. Flexibility enables robots to generate context-dependent behavior that is suitable for the current situation. As a complement, robustness enables robots to generate behavior without being affected by perturbations, such as unknown irregularities and unrelated or noisy sensory information. Flexibility can be realized by accepting sensory information about the current environment, and robustness by ignoring the information or assessing its relative importance. From this viewpoint, flexibility and robustness are conflicting demands, which makes it non-trivial to achieve both at the same time. We aim to tackle this issue in terms of predictive learning of sensorimotor

A. Lintas et al. (Eds.): ICANN 2017, Part I, LNCS 10613, pp. 11–18, 2017.
https://doi.org/10.1007/978-3-319-68600-4_2

information with uncertainty estimation, which has been widely accepted as a key computational principle for cognitive functions, including action, perception, and attention [1].

In the context of robot learning, Noda et al. [2] demonstrated that a small humanoid robot with a connectionist framework using a recurrent neural network with parametric biases (RNNPB) [3] can dynamically generate and switch its object-handling behavior. Their RNNPB was trained to predict sensorimotor information by receiving current information and integrating it with contextual information stored in the network. The key point is that the network received a mixture of actual and predicted sensorimotor states as network input. Modulation of network input is an important aspect of using RNNs. For example, in the studies [4,5], the performance of RNNs was improved by replacing the predicted states with the actual (true) states for network input in the training phase. In the study by Noda et al. [2], mixing the actual and predicted states together results in both flexibility in the face of environmental change and robustness against noise. However, the usefulness of this method as presented is limited because the mixture ratio must be hand-tuned for each target task, after which the tuned ratio was static through the task.

In the present study, we speculate that the uncertainty of sensory information can be used to dynamically modulate the input state of RNNs so as to realize flexible and robust robot behavior. Specifically, we propose a method in which the uncertainty of a future actual sensory state is estimated by a so-called stochastic continuous-time RNN (S-CTRNN) [6]. The estimated uncertainty is used as a factor in determining the mixture ratio between actual and predicted sensory states for the network input. The proposed method is validated by a robot-learning experiment that compares its results with those from a conventional method.

2 Computational Framework

2.1 Overview of S-CTRNN

S-CTRNN is an extension of conventional CTRNNs [7], and it consists of input, context, output, and variance layers. The distinguishing characteristic of this network is the newly added variance layer, which is used to estimate the uncertainty of target states. As a generative model, the network learns to predict the mean $\boldsymbol{y}_t$ and variance (uncertainty) $\boldsymbol{v}_t$ of a target state $\hat{\boldsymbol{y}}_t$ given the input state $\boldsymbol{x}_t$ and the context $\boldsymbol{c}_t$ stored in the network, where the target state at time step t characterizes the input state at the next time step $t+1$. The internal state of the ith neural unit at time step t ($u_{t,i}$) in each layer other than the input layer is described by

$$u_{t,i} = \begin{cases} \left(1 - \frac{1}{\tau_i}\right) u_{t-1,i} + \frac{1}{\tau_i} \left(\sum_{j=1}^{N_{\mathrm{I}}} w_{ij} x_{t,j} + \sum_{j=1}^{N_{\mathrm{C}}} w_{ij} c_{t-1,j} + b_i \right) & (i \in I_{\mathrm{C}}), \\ \sum_{j=1}^{N_{\mathrm{C}}} w_{ij} c_{t,j} + b_i \quad (i \in I_{\mathrm{O}} \cup I_{\mathrm{V}}), \end{cases} \tag{1}$$

where I_{C}, I_{O}, and I_{V} are the index sets for the context, output, and variance layers, respectively; N_{I}, N_{C}, N_{O}, and I_{V} are the numbers of the input, context, output, and variance units, respectively; $x_{t,j}$ is the jth input state at time step t; $c_{t-1,j}$ is the state of the jth context at time step $t-1$; τ_i is the time constant of the ith context unit; w_{ij} is the synaptic weight of the connection from the jth to the ith unit; and b_i is the bias of the ith unit. The activation state for the context and output states is computed by $\tanh(u_{t,i})$; that for the variance state is computed by $\exp(u_{t,i})$.

The goal of predictive learning is to maximize the likelihood $L(\boldsymbol{\theta})$, which is derived from the Gaussian assumption:

$$L(\boldsymbol{\theta}) = \prod_{t=1}^{T} \prod_{i=1}^{N_{\mathrm{O}}} \frac{1}{\sqrt{2\pi v_{t,i}}} \exp\left(-\frac{(\hat{y}_{t,i} - y_{t,i})^2}{2v_{t,i}}\right), \tag{2}$$

where $\boldsymbol{\theta}$ is a set of network parameters (w_{ij}, b_i), T is the length of the time series, and $\hat{y}_{t,i}$ is the ith target state corresponding to the next input state $x_{t+1,i}$. The network parameters are optimized by using the gradient ascent method with back-propagation through time (BPTT), as detailed in [6].

2.2 Mixing Actual and Predicted States Using S-CTRNN

Conventionally, forward computation of RNNs is performed via open-loop generation or closed-loop generation. In open-loop generation, shown in Fig. 1 (left), the input layer receives the actual state, such as recorded or online sensory data, at time step $t+1$, and the state is taken as the target state at the previous time step t ($x_{t+1,i} = \hat{y}_{t,i}$). In closed-loop generation, shown in Fig. 1 (center), in contrast, the input layer receives the output state generated at the previous time step t, which corresponds to the prediction of the input state at the current time step $t+1$ ($x_{t+1,i} = y_{t,i}$).

Here, we propose to mix these operations according to the variance predicted by the S-CTRNN, which represents the uncertainty of the next input state. This is done as follows:

$$x_{t+1,i} = (1 - \alpha(v_{t,i}))\, \hat{y}_{t,i} + \alpha(v_{t,i}) y_{t,i}, \tag{3}$$

where $0 \leq \alpha(v_{t,i}) \leq 1$ is the mixture ratio, represented by a monotonically increasing function of the uncertainty. This equation is derived from the idea that actual states with high uncertainty, which may perturb the network dynamics, should not be input to the network and should, instead, be replaced with predicted states for stability. When $\alpha(v_{t,i})$ is a fixed value that does not depend on the time step t and the element i, Eq. (3) corresponds to the method used in the study by Noda et al. [2]. The generation method, which can be called *mixture-loop* generation, is illustrated in Fig. 1 (right). In the figure, as an example, the S-CTRNN generates low uncertainty for the first dimension and high uncertainty for the second. The first dimension of the input layer at the next time step receives the actual state (which has a relatively certain estimate) in

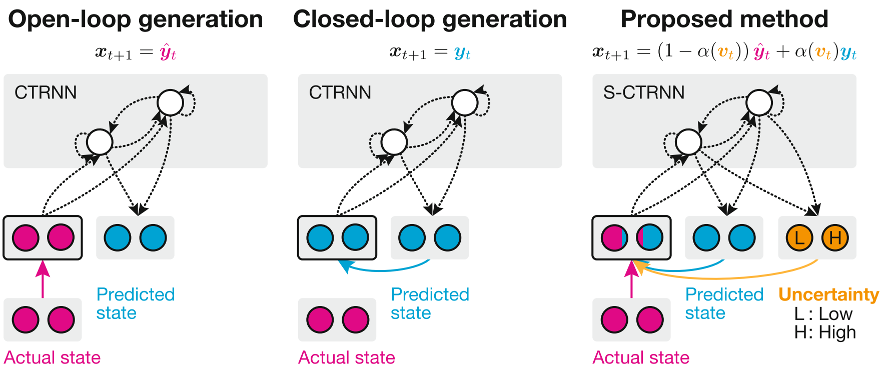

Fig. 1. Different generation methods. Left: open-loop generation with a conventional CTRNN in which the actual state $\hat{\boldsymbol{y}}_t$ (magenta) is fed into the input layer. Center: closed-loop generation with a conventional CTRNN in which the predicted state $\boldsymbol{y}_t$ (cyan), instead of the actual state, is fed into the input layer. Right: proposed method (mixture-loop generation) with an S-CTRNN in which the actual and predicted states are mixed according to the mixture ratio $\alpha(\boldsymbol{v}_t)$, where $\boldsymbol{v}_t$ is a vector representing uncertainty or variance (orange) as estimated by the S-CTRNN. (Color figure online)

greater proportion than the predicted state. In contrast, the second dimension receives the predicted state in greater proportion than the actual state (which has an uncertain estimate). This mixture method, which is specific to each time step and dimension, is expected to contribute to the flexibility and robustness of robot behavior.

3 Robot Experiment

3.1 Task Setting

We performed a robot learning experiment to evaluate the proposed method. In the experiment, a small humanoid robot “NAO” (Aldebaran Robotics) was used and interacted with a human experimenter, who was wearing a red glove to ease visual processing. The robot was required to perform an interactive task with the experimenter. The required task consisted of two phases: a sensor-dependent task and a sensor-independent task, as shown in Fig. 2.

In the first phase (the sensor-dependent task), the experimenter moved his hand to the left or right relative to the home position, and the robot was required to raise its corresponding hand. The direction of the human hand movement was randomly selected with equal probability by generating a sequence in advance. This phase was repeated until the experimenter cued a task transition by putting his hand up. Whether to continue the current task or transition to the other task was also determined with equal probability.

In the second phase (the sensor-independent task), after the demonstration of the transition cue, the robot was required to alternately raise its right and

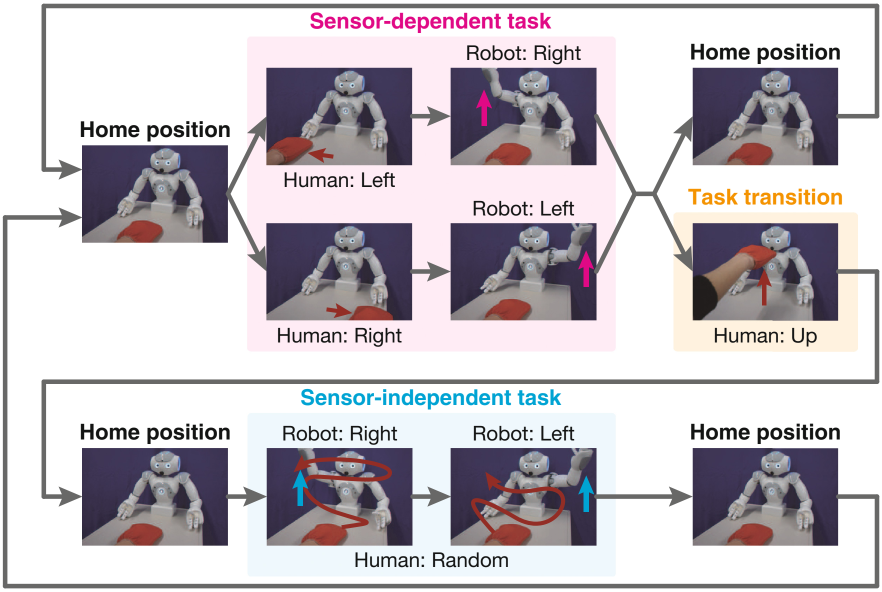

Fig. 2. Interactive task between a human experimenter and a small humanoid robot "NAO". The task has sensor-dependent and sensor-independent tasks.

left hands once, independent of the human hand movement. During this phase, the experimenter made hand movements as distractions, such as by randomly moving the hand or keeping the hand at a specific position. That is, in this phase, the robot needed to ignore unrelated sensory states (visual information from the human hand movement) and perform its own task. After finishing this phase, the task returned to the first phase again without giving any cue that task transition had occurred.

3.2 Experimental Procedure

The robot learning experiment consisted of three phases: recording of training data, training the S-CTRNN, and testing the performance of the robot. In what follows, each phase is briefly introduced.

In the first phase (data recording), training data were collected through kinesthetic teaching, with the robot controlled by directly guiding its arm movements. The recorded training data consisted of 3-dimensional visual information and 8-dimensional motor information. The visual information comprises time-series data of the center of gravity and area ratio of the red glove as extracted from a visual image obtained by a camera mounted on the robot. The motor information comprises time-series data of the four joint angles of each arm of the robot. In this data recording phase, the sensor-dependent task was repeated several times and then the task switched to the sensor-independent task. The entire procedure was repeated twice, and in total 10 training data were recorded, each of which included two transitions between the sensor-dependent and sensor-independent tasks.

In the second phase (network training), the S-CTRNN was trained offline by using the recorded data from the first phase. The numbers of input, context, output, and variance units were $N_{\mathrm{I}} = 11$, $N_{\mathrm{C}} = 55$, $N_{\mathrm{O}} = 11$, and $N_{\mathrm{V}} = 11$, respectively. The context units were divided into two groups, fast context units ($N_{\mathrm{FC}} = 50$) and slow context units ($N_{\mathrm{SC}} = 5$), on the basis of a time constant and connection setting to introduce the multiple timescale property proposed by Yamashita and Tani [8]. The time constants of the fast and slow context units were $\tau_{\mathrm{FC}} = 5$ and $\tau_{\mathrm{SC}} = 50$, respectively. The fast context units were connected with the input, fast context, slow context, output, and variance units. In contrast, the slow context units were connected with only the fast and slow context units to constrain the information flow. Details of the multiple timescale property derived from these settings are discussed in [8].

In the third phase (performance testing), the trained S-CTRNN was installed in the robot, and the actions generated via the conventional open-loop and proposed mixture-loop generation methods were compared. For the monotonically increasing function in Eq. (3), we used $\alpha(v_{t,i}) = v_{t,i}/v_{\max}$, where $v_{\max} = 0.01$ is a predefined parameter representing an upper bound of the estimated uncertainty. In this testing phase, the sensor-dependent task was repeated twice and then the processing was switched to the sensor-independent task. This set of procedures was repeated twice, meaning that, in all, each trial included four sensor-dependent tasks and two sensor-independent tasks.

4 Results and Discussion

In the sensor-dependent task, both the robot with open-loop generation and that with the proposed mixture-loop generation were able to perform flexible behavior corresponding to demonstrated human hand movements. However, in the sensor-independent task, only the robot with mixture-loop generation succeeded in generating learned behavior robustly; the robot with open-loop generation failed.

Examples of the time-series data during action generation with each method are shown in Fig. 3. In the sensor-dependent task, the experimenter first moved the hand to the right side and then to the left side. The network outputs show the corresponding movement of both the left and right robot arms. In the sensor-independent task with open-loop generation, although the robot needed to first raise its right hand and then its left hand, it moved the left arm first (represented by the gray ellipse). After this failure, the robot moved the left arm again for the latter part of the sensor-independent task. However, the movement was not enough large relative to the other movements. In contrast, no failures occur with the proposed mixture-loop generation: both the sensor-dependent and sensor-independent tasks are successfully completed. It should be noted that the ratio of actual states used for the network inputs dynamically changes at around time step 300, which corresponds to the moment after the task transition. This dynamic modulation of the mixture ratio between actual and predicted sensory

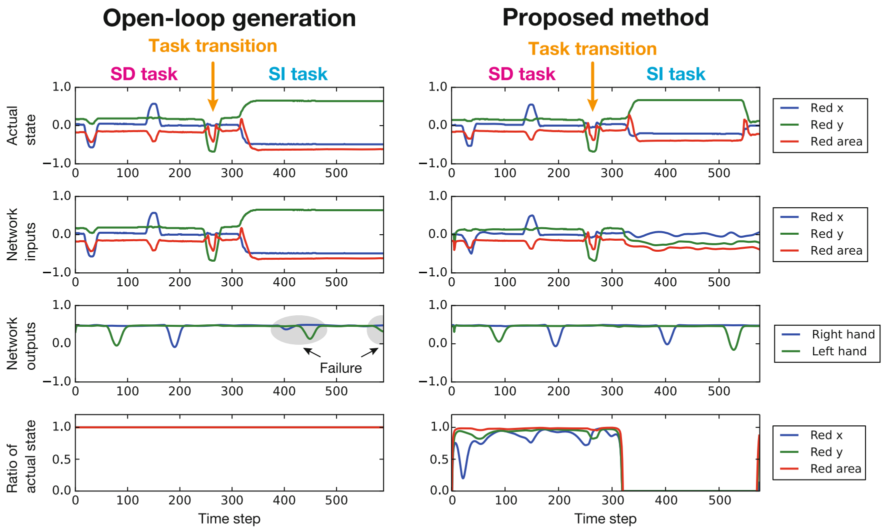

Fig. 3. Time-series data of actual states, network inputs, network outputs, and the ratio of actual states with open-loop generation and the proposed mixture-loop generation. Task transition points from the sensor-dependent (SD) task to the sensor-independent (SI) task are indicated by orange arrows. Failures are indicated by the gray ellipses in the sensor-independent task with open-loop generation.

states on the basis of estimated uncertainty enables the robot to perform flexible and robust behavior by accounting for necessary visual states in the sensor-dependent task and ignoring unnecessary visual states in the sensor-independent task.

We characterized the success of the proposed method by taking success rates for generated movements in both methods. Specifically, we set a threshold for the shoulder roll angles each necessary depending on the situation. If the angle value exceeded the threshold within a certain period, the generated movement was considered successful. Table 1 shows the success rates for generated movements with the conventional open-loop generation and with the proposed mixture-loop generation in each task. The results demonstrate that the proposed method outperforms the open-loop generation on both tasks.

Table 1. Success rates for generated movements out of 160 trials

	Open-loop generation		Proposed method	
	SD task	SI task	SD task	SI task
Number of successes	145	106	157	154
Success rates	0.906	0.663	0.981	0.963

5 Conclusions

In this study, we proposed a method to dynamically modulate the input state of RNNs in order to realize flexible and robust robot behavior. We employed S-CTRNN as a computational framework, estimating the uncertainty of next sensory states and then using the estimated uncertainty to determine the mixture ratio between actual and predicted states for the next network input. We performed a robot learning experiment to evaluate our proposed method. The task for the robot consisted of a sensory-dependent task and a sensory-independent task. The former task required the robot to incorporate meaningful visual information, and the latter required the robot to ignore meaningless visual information. The experimental results demonstrated that our proposed method enabled the robot to behave flexibly and robustly. Future work will focus on applying the proposed method to more practical tasks and on extending the method to high-dimensional sensory data, such as raw visual images by using state of the art data sets, instead of only low-dimensional feature information.

Acknowledgement. This work was supported in part by JST CREST Grant Number: JPMJCR15E3, Japan;JSPS KAKENHI Grant Numbers: 25220005, 17K12754, Japan and the “Fundamental Study for Intelligent Machine to Coexist with Nature” program of the Research Institute for Science and Engineering at Waseda University, Japan.

References

1. Clark, A.: Surfing Uncertainty: Prediction, Action, and the Embodied Mind. Oxford University Press, Oxford (2015)
2. Noda, K., Ito, M., Hoshino, Y., Tani, J.: Dynamic generation and switching of object handling behaviors by a humanoid robot using a recurrent neural network model. In: Nolfi, S., Baldassarre, G., Calabretta, R., Hallam, J.C.T., Marocco, D., Meyer, J.-A., Miglino, O., Parisi, D. (eds.) SAB 2006. LNCS, vol. 4095, pp. 185–196. Springer, Heidelberg (2006). doi:10.1007/11840541_16
3. Tani, J., Ito, M.: Self-organization of behavioral primitives as multiple attractor dynamics: a robot experiment. IEEE Trans. Syst. Man Cybern. - Part A Syst. Hum. **33**(4), 481–488 (2003)
4. Williams, R.J., Zipser, D.: A learning algorithm for continually running fully recurrent neural networks. Neural Comput. **1**(2), 270–280 (1989)
5. Bengio, S., Vinyals, O., Jaitly, N., Shazeer, N.: Scheduled sampling for sequence prediction with recurrent neural networks. arXiv 19994575, pp. 1–9 (2015)
6. Murata, S., Namikawa, J., Arie, H., Sugano, S., Tani, J.: Learning to reproduce fluctuating time series by inferring their time-dependent stochastic properties: application in robot learning via tutoring. IEEE Trans. Auton. Ment. Dev. **5**(4), 298–310 (2013)
7. Doya, K., Yoshizawa, S.: Adaptive neural oscillator using continuous-time backpropagation learning. Neural Netw. **2**(5), 375–385 (1989)
8. Yamashita, Y., Tani, J.: Emergence of functional hierarchy in a multiple timescale neural network model: a humanoid robot experiment. PLoS Comput. Biol. **4**(11), e1000220 (2008)

http://www.springer.com/978-3-319-68599-1

Artificial Neural Networks and Machine Learning – ICANN 2017
26th International Conference on Artificial Neural Networks, Alghero, Italy, September 11-14, 2017, Proceedings, Part I
Lintas, A.; Rovetta, S.; Verschure, P.F.M.J.; Villa, A.E.P. (Eds.)
2017, XXXI, 469 p. 191 illus., Softcover
ISBN: 978-3-319-68599-1