A Brain-Inspired Approach for Probabilistic Estimation and Efficient Planning in Precision Physical Interaction

Dengpeng Xing, Yiming Yang, Tielin Zhang, and Bo Xu

***Abstract*—This paper presents a novel structure of spiking neural networks to simulate the joint function of multiple brain regions in handling precision physical interactions. This task desires efficient movement planning while considering contact prediction and fast radial compensation. Contact prediction demands the cognitive memory of the interaction model, and we novelly propose a double recurrent network to imitate the hippocampus, addressing the spatiotemporal property of the distribution. Radial contact response needs rich spatial information, and we use a cerebellum-inspired module to achieve temporally dynamic prediction. We also use a block-based feedforward network to plan movements, behaving like the prefrontal cortex. These modules are integrated to realize the joint cognitive function of multiple brain regions in prediction, controlling, and planning. We present an appropriate controller and planner to generate teaching signals and provide a feasible network initialization for reinforcement learning, which modifies synapses in accordance with reality. Experimental results demonstrate the validity of the proposed method.**

***Keywords*—Brain-inspired structure, precision physical interaction, spiking neural networks.**

I. Introduction

In precision physical interaction, robots contact the environment [1], [2] when executing high-precision tasks and perform an appropriate strategy via the conveyed force information to improve efficiency while alleviating undesired effects. Among all related works involving physical interactions, one kind is interesting where the controlled object axially moves and radially contacts a passive counterpart. For example, in peg-in-hole assembly [3], slippage during grasping [4], digging, and drilling [5], radial contacts are undesired but indispensable as axial movement proceeds, and at a mesoscale or microscale level, they generate substantial influence. Appropriate controllers and efficient planners are crucial in accelerating axial movements while keeping radial contacts under control.

Different from the physical human-robot interaction widely studied in robotics, physical interactions investigated in this paper mainly involve the contacts between objects during tasks. This interaction type dominates in the execution process, especially for small objects, and is explored through modeling, estimating, and compensating [6] for performance improvement. In precision assembly, the robot inserts a peg into a hole and, due to many factors such as uncertainty and misalignment, this movement causes radial contact forces, which need to be well handled, especially for fast insertion. Experimental data indicate that this process conforms to a Gaussian distribution [7], based on which the prediction and compensation show its power in trajectory planning. Physical interaction is also an essential aspect in robot grasping: friction can improve the grasp robustness while rendering it challenging to place fingers precisely on the object's perimeter [8]. Classic robot solutions are usually able to solve common interaction problems [9], but if looking into the effortlessness of creatures in coping with environments, we expect to learn from those biological models.

As categorized by Maass [10], spiking neural networks (SNNs) are the third generation of neural networks, exchanging information via spikes [11], [12]. They address the nature of the brain [13], and their main advantages originate from brain imitation, incorporating the processing ability of spatiotemporal signals and the event-based property that contributes to energy efficiency [14]. Biological intelligence can achieve functions, with high efficiency and adaptation, of perceiving, memorizing, and thinking, and it inspires researchers to design similar structures to grant robots the ability to mimic the underlying mechanisms of the brain [15]. Robot control aims at problems of a time series evolution of dynamics and physical interactions, and the coding in SNNs captures the temporal properties of sensorimotor signals. These networks are intrinsically suitable for robot control [16], [17]. SNNs have contributed to many types of robots in solving various tasks and have achieved substantial performance. However, applying the merits of SNNs to solve the problems in precision physical interaction remains open. Another question is "can we investigate the multiple functions of the brain when accurately operating?"

This paper presents a structure to imitate the joint function of multiple brain regions in dealing with precision physical interaction tasks. To simulate the memory function of the hippocampus, we propose a double recurrent network with the spatiotemporal characteristic of distribution to predict radial contact. A cerebellar module is applied to emulate the temporally dynamic prediction of radial contact forces. To mimic the cognitive planning of the prefrontal cortex, we adopt a block-based feedforward network to plan movements with current and predicted contacts. This paper integrates these modules to realize the joint cognitive function of multiple brain regions. We design an appropriate controller and planner to generate teaching signals, by which the processing of each module is also depicted. We then apply reinforcement learning to approximate optimal performance. We carry out experiments to validate the proposed method.

Manuscript received Jul. 19, 2021; revised Dec. 08, 2021; accepted Mar. 28, 2022. This work was supported by the National Nature Science Foundation of China under Grant (62073324) and by the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant (XDA27010404). (Corresponding author: Dengpeng Xing)

Dengpeng Xing, Yiming Yang, Tielin Zhang, and Bo Xu are with the Institute of Automation, Chinese Academy of Sciences, Beijing, and with the School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China.

To our knowledge, this is the first address of applying SNNs for efficient planning and contact handling in precision physical interaction tasks, and the contributions are listed below:

- We propose a novel structure to imitate the joint function of multiple brain regions in handling high-precision tasks, in which prediction, contact, and compensation are involved. This structure simulates the memory function of the hippocampus, the controlling function of the cerebellum, and the planning function of the prefrontal cortex.
- We present a double recurrent network to approximate the spatiotemporal property of distribution. In this network, the outer recurrence is at the network level, fulfilled by an output feedback loop, and the inner recurrence is at the neuron level, realized by a recurrent SNN.

Our motivation is to map control functions to different functional brain regions by using a biologically plausible SNN to obtain better interpretability. Different sub-networks can be coordinated in the spirit of multi-brain cooperation in the biological brain, attempting to make further integrative research easier by bridging traditional control theories and new biological foundations.

II. Related Work

A. Precision physical interaction

Tasks involving precision physical interactions require much attention on contact control [18], and model-based and model-free controllers are primarily investigated. The model-based controller has a predetermined mathematical model to describe the relationship between the misalignment and the object contact and considers the mismatch between the model and the real contact. A variety of models are built according to shapes and materials, e.g., using Gaussian models [7] to describe the uncertainty, modeling with a set of contact states [19], learning admittance control with Gaussian mixture regression [20]. Model-based controllers usually show efficiency due to the consideration of prior knowledge. Building models may be constrained to specific tasks, and model-free controllers adjust positions only based on the measured forces. A fixed step was used in [21] to handle insertion with interference fit, and step scaling with radial forces was adopted in [3] for sequence assembly. It is a conservative solution but gains wide applications. Examining how the creatures interact with the environment yields the importance of learning from biology, but this problem remains unresolved.

B. Brain-inspired robot control

In recent years, many brain-inspired approaches have been applied in robotics and control [22]. Information in a biological nervous system is transmitted through spike electric currents, which are sufficient to drive learning and behavior [23], and SNNs aim to imitate this mode of information transmission. Unlike traditional artificial neurons, spiking neurons can take advantage of the spatiotemporal information of the signals and are expected to process continuous inputs from the dynamic world [14]. Another distinction of SNNs is the event-driven nature, which is the foundation of energy efficiency. However, due to the non-differentiability of spikes, the backpropagation algorithm cannot be used directly for training. There are two main categories to address the learning of SNNs. One is the conversion-based approach, which trains traditional neural networks and then transfers them to SNNs [24], [25]. This category achieves the most competitive accuracy on large-scale SNNs [14]. The other is the spike-based approach, in which SNNs are trained by timing information directly [26]. It gains sparsity and efficiency in terms of neuronal dynamics.

Applying SNNs to robot control problems has achieved satisfactory performance. For instance, they can map the environment in unidimensional simultaneous localization and mapping [27], plan a path in complex natural environments with altered axonal delays [28], handle compliant control [29], and predict linear, nonlinear, and chaotic dynamics [30]. A few methods implement SNNs to solve physical interaction tasks, and a recent example is that they were used to trigger robot finger motion to interact with human hands based on human surface electromyography data [31].

In addition to the neuronal dynamics, the imitation of brain regions and their coordination have also been investigated. In [32], a cerebellar-like SNN controller was proposed to address delays in human-robot interaction. A multi-brain region model containing millions of neurons was established to simulate human vision and response pathways [33]. Inspired by the function of different brain regions, researchers proposed the FORCE method for supervised learning in SNNs [34]. However, in the field of precise physical interaction, few studies attempt to imitate the coordination mechanisms of multiple brain regions.

III. The task model and the method overview

We briefly introduce the properties of the task and describe the outline of the proposed structure.

A. Task description

This paper considers motion planning and control where physical contact is primarily concerned. The controlled object moves along a primary direction with a velocity $\dot{m}_a$ and receives a radial contact force $\boldsymbol{f}_r$ from its counterpart, which relates to the movement velocity. To fully understand it, an example is the precision assembly, where the variation of the radial force of peg-in-hole is found via experiments positively correlating to the insertion depth. This type of task requires an efficient plan of axial movement while deliberately considering radial contact. We restrict our research to the condition where the variation in contact forces conforms to Gaussian distributions. We take the movement as the controllable variable and radial contact as the state, and the probability of the force variance results in

$$P\left(\dot{\boldsymbol{f}}_r|\dot{m}_a\right) = \mathcal{N}\left(\boldsymbol{\mu}_{\dot{m}_a}, \boldsymbol{\Sigma}_{\dot{m}_a}\right), \tag{1}$$

where $\boldsymbol{\mu} = [\mu_x, \mu_y, \mu_z]^{\mathrm{T}}$ is the vector representing the radial contact expectation,

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_x^2 & \sigma_{x,y} & \sigma_{x,z} \\ \sigma_{x,y} & \sigma_y^2 & \sigma_{y,z} \\ \sigma_{x,z} & \sigma_{y,z} & \sigma_z^2 \end{bmatrix}$$

is the variance matrix, and the subscript $\dot{m}_a$ shows that the corresponding variable is related to the primary movement. Label a vector $\boldsymbol{\sigma}^2$ to include all elements in $\boldsymbol{\Sigma}$ without repetition. The Gaussian parameters scale with movement velocity and their relationship is supposed to be known in advance. The above equation indicates that the radial force variation depends on the primary movement. To view it from another perspective, we rewrite it in the time sequence format

$$P\left(\boldsymbol{f}_{r,t+1} | \boldsymbol{f}_{r,t}, \int_t^{t+1} \dot{m}_a dt\right) = \mathcal{N}\left(\boldsymbol{f}_{r,t} + \boldsymbol{\mu}_{t+1}, \boldsymbol{\Sigma}_{t+1}\right). \quad (2)$$

We replace the subscript of distribution parameters from $\dot{m}_a$ to $t+1$ to emphasize the time sequence of contact variation other than its dependence on movement. We label $\dot{m}_{max}$ and $\ddot{m}_{max}$ the limitations of the object's velocity and acceleration.

An efficient policy intends to speed up the movement, which will induce a larger radial force, and this force demands to slow down for corresponding compensation. This reflects the importance of efficient planning in considering these contradictory influences. If taking the rules of the variation of radial forces into consideration, we can plan efficiently by coordinating movements and radial forces. The task is clearly defined to devise the movement velocity while maintaining an as-small-as-possible radial force under the condition of uncertainty

$$\begin{aligned} \dot{\boldsymbol{m}}_a = \operatorname{argmin} \sum_{t=1}^{T} \left[\left(l - \int_0^t \dot{m}_{a,t} dt \right)^2 + \gamma_f \boldsymbol{f}_{r,t}^{\mathrm{T}} \boldsymbol{f}_{r,t} \right], \\ \text{s.t.} \quad \dot{m}_{a,t} \leq \dot{m}_{max}, \quad \ddot{m}_{a,t} \leq \ddot{m}_{max} \end{aligned} \quad (3)$$

where l is the total length to go, T is the total counted time, γ_f is the weight of the radial force relative to the unattained distance, and $\dot{\boldsymbol{m}}_a$ is the set of $\dot{m}_{a,t}$ with $t = 1, \cdots, T$.

This task and the above three equations require at least three modules to achieve good performance. One is to store the parameters of Gaussian distributions, which illustrates the possibility of future contact; one aims to compensate for the radial contacts smoothly and efficiently; the other is to plan the primary velocity considering the outcomes of the other two modules.

B. Overview of the approach

As shown in Fig. 1, the brain-inspired networks proposed in this paper incorporate three parts: the hippocampus, the cerebellum, and the prefrontal cortex. The hippocampus is associated primarily with memory; the cerebellum coordinates sensory input with muscular responses; the prefrontal cortex is responsible for orchestrating thoughts and actions in accordance with internal goals.

The reasons that these three brain regions are imitated lie in the property of the task. Radial contact forces need to be compensated promptly, and a primary movement results in the emergence or augmentation of radial contact forces, as addressed in Eq. 1. We first need a controller to compensate for any radial contact forces. Since we already know that the contact process conforms to Gaussian distributions, predicting future contact based on primary movements and the compensation controller is preferable for efficient planning. That demands a prediction mechanism. In conditions of uncertainty, it is inappropriate for motion planning to consider only the current state, and on the contrary, we suitably take into account previous performance, current assessment, and future prediction. This comprehensive consideration covers different factors that affect planning performance and calls for a complicated planner. All these in sequence require functions of immediate compensation, distribution memory, and decisive planning, and their interactions are crucial.

Viewing the above analysis, we mimic the hippocampal network to store the contact model, with which future states are predicted, i.e., outputting radial contact states of the next several steps given the current movement. Since the prediction outcome conforms to a probabilistic model, this hippocampal network needs to learn the distribution. The cerebellar network acts as a controller, compensating whenever radial forces exist. It considers the current and predicted contacts and results in compensational movements and altered predicted values. The hippocampal and cerebellar networks are united in the contact prediction of the next several steps. This unity regards the compensational capability in predicting, which is closer to reality.

The prefrontal cortex network serves as a planning module, determining the movement magnitude to execute. Based on past performance (inner recurrence), the current state (tactile feedback), and the predicted contacts (from the other two brain regions), this part outputs the movement to be performed and updates an internal parameter. The network unity of the prefrontal cortex and the cerebellum plays the role of a policy generator, producing movement commands. The control signal is sent to drive the platform, which advances the process of goal approaching and changes the physical contact, and the force sensor measures the new force and feedbacks it to the networks.

In this network structure, the cerebellum participates twice in one control period, correlating with the hippocampus in prediction and independent planning from the prefrontal cortex in control. From the viewpoint of the time sequence, the lower modules in Fig. 1, including the prefrontal cortex and the cerebellum, calculate the movement at the current time with real-time force feedback. The upper brain regions incorporating the hippocampus and the cerebellum perform predictions in the future horizon. We employ each brain region based on the requirement of motion planning with distinct functions and present the structure of each part referring to the findings in neuroscience.

In Fig. 1, each module is surrounded by squares in colors and is correlated to the corresponding brain region by a purple line. The red dashed line separates the actuators and the regulation part. This line also depicts the process of encoding and decoding between the measured continuous forces and spike trains sent to the network. In other words, above the

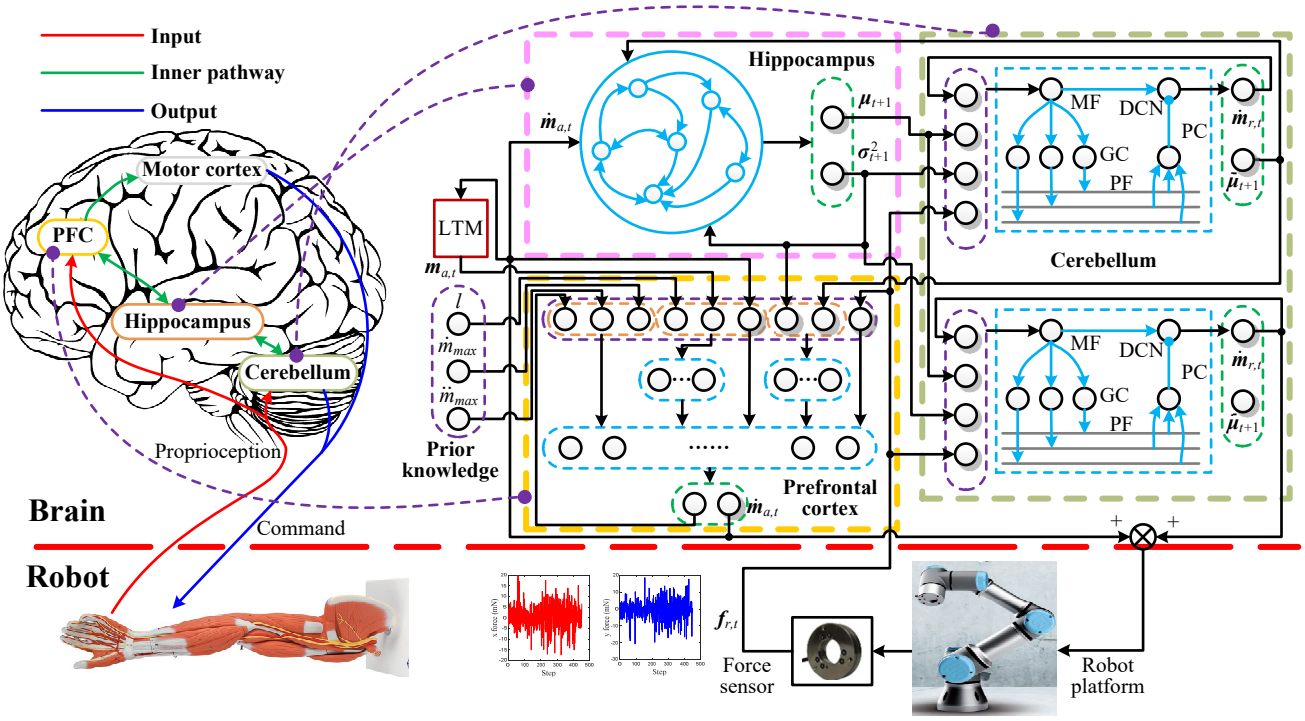


Fig. 1. General illustration of the brain-inspired network. The left schematically shows the pathway of how the brain and the arm cooperate in operation. The right panel shows the networks simulating brain regions and the information flow to and from the platform. Above the red dashed line are the brain and its simulated network. The robot platform and its arm role are shown below. Colored squares isolate networks mimicking different brain regions, and their relations are connected by purple curves. Note that there is only one module in the cerebellum, but since in the whole structure it is associated with the prefrontal cortex and the hippocampus, we mirror it to clearly show the connections. The acronyms include LTM (long-term memory), MF (mossy fiber), GC (granule cell), PF (parallel fiber), PC (Purkinje cell), and DCN (deep cerebellar nuclei).

red dashed line are computed in spikes, and below is executed in rational numbers.

This paper imitates the brain from three aspects. At the functional level, we imitate the coordination of multiple related brain regions based on their primary functions, mimicking the pathways in the brain. The connection between the hippocampus and the cerebellum predicts what future contacts will be. The union of the prefrontal cortex and the cerebellum plans the current action. At the circuit level, we also design the network structure of each module according to its properties. A double recurrent network is proposed to simulate long-term to short-term memory; a feedforward network is adopted to imitate the prefrontal cortex; a cerebellar-like structure is adopted. At the neuron level, we employ SNNs to mimic the neuronal dynamics of the brain.

IV. The Brain-Inspired Networks

We introduce the basics of the spiking neurons, describe each module in detail, including structures, synaptic modifications, and specific properties, and discuss the biological plausibility.

A. Processing of the spiking neurons

Spiking neurons with synapses play an essential role in mimicking brain functions of memory storage and data processing. The leaky integrated-and-fire (LIF) model is commonly adopted with simplified neuron dynamics

$$\tau_m \frac{dU}{dt} = -[U - U_{rest}] + RI \tag{4}$$

where U is the membrane potential, U_{rest} is the resting potential, τ_m is the membrane time constant, I is the input current, and R is the membrane resistance. It uses a linear differential equation to describe the integration of a passive membrane. A neuron fires a spike and resets the potential to U_{rest} when its accumulated membrane potential surpasses the threshold V_{th}. This paper adopts an iterative LIF model [35] to simulate the neuron dynamics

$$U_{t+1} = k_r U_t (1 - S_{o,t}) + W S_{i,t+1} \tag{5}$$

$$S_{o,t+1} = f(U_{t+1} - V_{th}) \tag{6}$$

$$f(x) = \begin{cases} 0, & x < 0 \\ 1, & \text{otherwise} \end{cases} \tag{7}$$

where S_i and S_o are the received and output spike trains and k_r is a decay factor.

We need to train the three modules independently and, in each input layer, employ population uniform encoding to generate spike trains. These modules are connected in joint training and experimental implementation, and the encoding method applies to transform the measured continuous force to

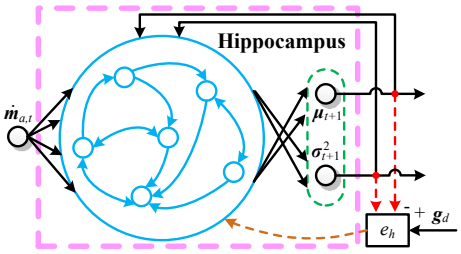


Fig. 2. The hippocampal network. The dotted red and orange lines indicate that the information is sent to the cost function and update the weights. Inside the blue circle is the RCSNN.

the input spike trains of the whole structure. We have attempted several other encoding methods, e.g., time-to-first-spike coding and Poisson coding, and found that this population uniform encoding is appropriate for this task from the viewpoint of repeatable accuracy. To further improve the encoding accuracy, we augment spiking neurons for each parameter in the input layer based on the range. We give an example to view this accuracy computation. One employs two hundred neurons to represent a variable ranging ± 0.5 mm. These neurons can fire up to ten thousand spikes within a 50 time-step training window, and filling the variable range into the maximum spike train yields a minimum resolution of 0.1 μm. More neurons improve the encoding accuracy and require more network parameters, resulting in the training burden. We need to consider this tradeoff in balancing the training difficulty and the network accuracy. The decoding process transforms spike trains of the whole structure's outputs into a command signal to drive the robot. We employ a rate-based method, in which we sum the spike number in a 50 time-step window (as used in the encoding example) and multiply it by the minimum resolution. More neurons in the output layer render higher transformation resolution.

Supervised learning and reinforcement learning are selectively employed in the module training. Spatio-temporal backpropagation (STBP) [35] is applied as a supervised learning method for network training with given teaching signals. For reinforcement learning, we use proximal policy optimization by changing the actor-network to SNNs.

B. The hippocampal network

We design the network shown in Fig. 2 to imitate the memory function of the hippocampus. This module attempts to learn the Gaussian distribution, as depicted in Eq. 1, and by doing so, we can consider the spatiotemporal characteristics of the contact prediction. The module input is the current primary movement $\dot{m}_{a,t}$, connected to all neurons of the subsequent recurrently connected SNN (RCSNN). Each neuron of this recurrent network is connected to the output layer via a one-to-all pattern. This module outputs the distribution parameters, indicating the next possible contact, $\boldsymbol{\mu}_{t+1}$ and $\boldsymbol{\sigma}^2_{t+1}$.

There are two reasons for picking the recurrent network in distribution learning. From a neuroscience point of view, the hippocampus is composed of recurrent networks. From the computational perspective, the task requirement in Gaussian learning spans a space where diverse primary movements correspond to different distribution parameters. This property demands the capability of processing a series of spatiotemporal information, and the recurrent spiking network is naturally suitable. Since we aim to predict future contacts, only a network with one recurrence is not enough to approximate Gaussian distributions of n steps later. Therefore, we propose a double-recurrent network to strengthen the temporal encoding in learning distribution parameters. The RCSNN is the inner recurrence, and the outer recurrence is the feedback loop of sending outputs into the RCSNN. It has both neuron-level and network-level recurrences, and combining these two levels leads to good performance in learning the essential property of Gaussian distribution. This novel structure can consolidate the relationship between each prediction step.

We may choose many types of RCSNNs in the inner recurrence depending on which is preferable, modeling accuracy or biological plausibility. The liquid state machine (LSM) [36] is a biologically plausible network, including a liquid layer. Neurons in this liquid layer are recurrently connected to capture dynamic information, which fades out over time. One of its peculiarities lies in that all the synaptic connections in the liquid layer are randomly initialized and fixed, avoiding vanishing gradients. The spiking neural unit (SNU) [37] addresses the dynamics of a spiking neuron from the viewpoint of recurrent neural networks, and as a result, training techniques, such as backpropagation, for artificial neural networks can be directly applied. We use this SNU as every single neuron and connect them in a recurrent manner, called RSNU. This structure can adjust its weights using BPTT and achieve good performance. In experiments, we compared them, along with the difference in incorporating inner and outer recurrence.

Based on Eq. 1, we can acquire successive predictions by applying the convolution of multiple Gaussian distributions (see Appendix A for more details). To capture the temporal property, the network outcomes at time $t+2$ relate to those at time $t+1$

$$\boldsymbol{\mu}_{t+2} = 2\boldsymbol{\mu}_{t+1}, \quad \boldsymbol{\sigma}^2_{t+2} = 2\boldsymbol{\sigma}^2_{t+1}. \tag{8}$$

The relationship of the above equation also applies to a finite step n. We expect the network to learn both one-step Gaussian distribution parameters and the linear superposition property in the multi-step prediction. Therefore, we add the temporal convolution into the network and define the hippocampal error function

$$e_h = \sum_{i=1}^{n} \gamma_h^{i-1} \left(\boldsymbol{g}_i - 2^{i-1}\boldsymbol{g}_d\right)^{\mathrm{T}} \boldsymbol{Q}_h \left(\boldsymbol{g}_i - 2^{i-1}\boldsymbol{g}_d\right), \tag{9}$$

where e_h and $\gamma_h \in (0,1]$ are the error function and a discounted parameter in the hippocampal network, $\boldsymbol{g} = \left[\boldsymbol{\mu}^{\mathrm{T}}, \left(\boldsymbol{\sigma}^2\right)^{\mathrm{T}}\right]^{\mathrm{T}}$ is the parameter vector composed of the network outputs, $\boldsymbol{g}_d$ is the distribution vector consisting of the desired parameters of one-step prediction, and the matrix $\boldsymbol{Q}_h \in \mathbb{R}^{4\times 4}$ weights the variance errors relative to the mean errors. The above equation considers n future prediction steps, and the discounted parameter γ_h depicts the importance of one step over its previous step in the overall performance. The training samples

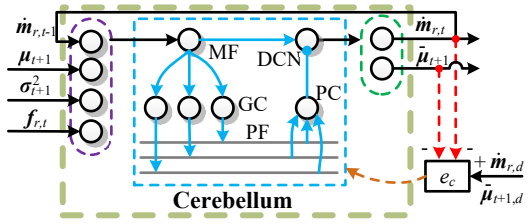


Fig. 3. The cerebellar network where the dotted red and orange lines send information to the cost function and to update the weights.

are acquired via experiments, and each $\dot{m}_a$ corresponds to a distribution vector $\boldsymbol{g}_d$. We backpropagate the error e_h to update the network weights.

C. The cerebellar network

Fig. 3 shows the network imitating the structure and function of the cerebellum. This module attempts to learn the mechanism of contact compensation. Collaborating with the hippocampus can deal with not only the current contact but also the future contact. The network inputs include two parts: the previous radial velocity $\dot{\boldsymbol{m}}_{r,t-1}$, which depicts the motion state of the robot, and the current radial contact $\boldsymbol{f}_{r,t}$ and the future one, including the predicted mean $\boldsymbol{\mu}_{t+1}$ and the variance $\boldsymbol{\sigma}^2_{t+1}$ to represent the prediction confidence. The outputs consist of the radial velocity and the contact residue prediction $\bar{\boldsymbol{\mu}}$. The velocity input is based on the requirement of continuous movement and the acceleration limitation. The velocity output intends to reduce the total radial contacts, which sums the current and the predicted one, considering the variance. The predicted contact residues estimate the remaining force after executing the output velocity. This estimation requires a learned model relating the radial movement to its contact variation. The contact residues are useful in predicting a series of future contacts uniting with the hippocampus and sending the next-step contact prediction to the prefrontal cortex.

The first layer consists of mossy fibers (MFs), which propagate the received information to both the granule cell (GC) layer and the deep cerebellar nuclei (DCN) layer. Modeled as a state generator, GCs generate somatosensory neural activity according to their inputs, and each GC connects to four neurons [38] in the MFs. Parallel fibers (PFs) establish connections between GCs and Purkinje cells (PCs). The DCN receives two different inputs, inhibitory from the PCs and excitatory from the MFs, and fully connects to all neurons in the output layer.

There are two main types of afferent fibers in the cerebellum: mossy fibers and climbing fibers. Mossy fibers originate from the spinal cord and brain stem, and the information from each mossy fiber is sent to a large number of Purkinje neurons. Climbing fibers originate from the inferior olivary nucleus, and each Purkinje neuron receives inputs from only a single climbing fiber. Based on this difference in connection type and information source, we choose mossy fibers as input.

The fact that outputs attempt to reduce contact forces and predict force residues naturally requires a contact model, which depicts the relationship between elastic deformation and contact force. We obtain this model via many experiments and,

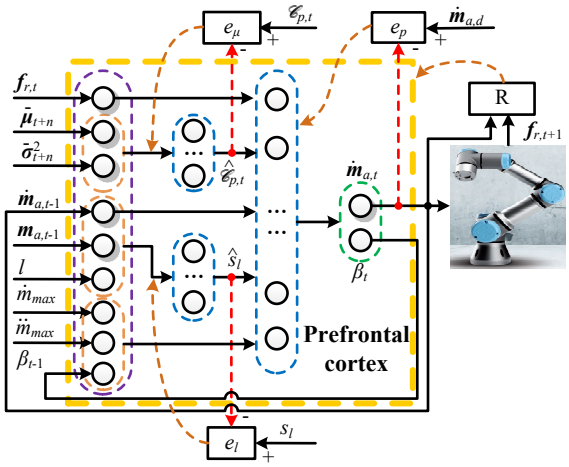


Fig. 4. The prefrontal cortex network where the cost functions receive the network outputs via dotted red lines and update the synapse via dotted orange lines.

based on it, design a compensational controller (see Section V-A) to generate teaching signals. The error function of the cerebellar network is composed

$$\begin{aligned} e_{c,t} = &\left(\dot{\boldsymbol{m}}_{r,t} - \dot{\boldsymbol{m}}_{r,d}\right)^{\mathrm{T}} \left(\dot{\boldsymbol{m}}_{r,t} - \dot{\boldsymbol{m}}_{r,d}\right) \\ &+ \gamma_c \left(\bar{\boldsymbol{\mu}}_{t+1} - \bar{\boldsymbol{\mu}}_{t+1,d}\right)^{\mathrm{T}} \left(\bar{\boldsymbol{\mu}}_{t+1} - \bar{\boldsymbol{\mu}}_{t+1,d}\right), \end{aligned} \tag{10}$$

where γ_c is the weight of the prediction error relative to the movement error, e_c is the error function in the cerebellar network, and the subscript d indicates that the corresponding variables are the desired variables. The above function considers both the responding action and the predicted contact after response. We acquire the training samples through the following process. We first experiment to obtain some key parameters, such as distribution parameters and object stiffness, and then generate a variety of data using the controllers discussed in Section V-A. We update the synaptic efficacy using supervised learning techniques to approximate the planned behavior.

D. The prefrontal cortex network

Fig. 4 shows the block-based feedforward network imitating the planning function of the prefrontal cortex. This module attempts to learn the mechanism of adapting movement based on contact forces and the task process. It collaborates with the prediction part (the hippocampus and the cerebellum) and orchestrates actions (the primary movement) in accordance with internal goals (achieving the target properly).

The network inputs include three segments: the contact force, the task process, and the inner parameter. The first segment consists of the current-measured radial force $\boldsymbol{f}_t$ and the predicted radial force; the second relates to the distance issues, including the desired length l, the attained distance $m_{a,t-1}$, the previous velocity $\dot{m}_{a,t-1}$, and the movement limitation; the last is β_{t-1}, representing the effect of past performance. We add physical meanings to reduce the computational burden. The predicted force is received from the other two brain

regions and is composed of the mean $\bar{\boldsymbol{\mu}}_{t+n}$ and variance $\bar{\boldsymbol{\sigma}}^2_{t+n}$. We send them to a block of isolated neurons in the second layer to separately compute their effects from the viewpoint of distribution comparison and label its output as $\mathscr{C}_{p,t}$. The inputs relating to distance incorporate the terminal signal, which affects movement selection, and we set aside another block of isolated neurons in the second layer for individual manipulation, with its output labeled as s_l. As a planning module, the prefrontal cortex network accepts comprehensive information about past performance, current evaluation, and future prediction to function effectively. We send these elements into the layers after blocks, in which the inner parameter β_t represents the past performance, the current force and velocity reflect the evaluation of the current state, and the predicted results depict future possibilities. These layers also accept the output of the distance-related block to arrive at the target appropriately. The network outputs include the planned primary movement and the updated internal parameter. The first output moves the object toward the target, and the second output is for successive planning, indicating previous performance-related information.

Similar to the training process of the cerebellum, we design a planning algorithm, as shown in Section V-B, to generate teaching signals. The error functions of different sub-modules are

$$e_\mu = \left(\hat{\mathscr{C}}_{p,t} - \mathscr{C}_{p,t}\right)^2, \quad e_l = \left(\hat{s}_l - s_l\right)^2, \tag{11}$$

$$e_p = \left(\dot{\boldsymbol{m}}_{a,t} - \dot{\boldsymbol{m}}_{a,d}\right)^{\mathrm{T}} \left(\dot{\boldsymbol{m}}_{a,t} - \dot{\boldsymbol{m}}_{a,d}\right), \tag{12}$$

where e_μ, e_l, and e_p are the error functions, e_μ dominates in the computation block of predicted forces, e_l regulates the outcome of the terminal signal block, e_p reflects the goodness of the overall output, $\mathscr{C}_{p,t}$ and s_l are the teaching signals for the two blocks, $\hat{\cdot}$ represents the outputs of each block, and $\dot{\boldsymbol{m}}_{a,d}$ is the desired velocity. We generate training samples by simulating various situations, given the obtained distribution parameters and object stiffness via experiments. We first update the synaptic efficacy of the two blocks independently and then tune the other part of the network, fixing the block-related weights. After the network is coarsely trained, we connect all modules and specifically tune the prefrontal cortex network utilizing reinforcement learning with the following reward

$$R = (-1)^{f\left(\left(l - \int_0^t \dot{m}_{a,t} dt\right)^2 - \varepsilon\right)} - \gamma_f \boldsymbol{f}_{r,t}^{\mathrm{T}} \boldsymbol{f}_{r,t}, \tag{13}$$

where ε is a threshold determining that the desired position is close and R is the one-step reward. Iteratively computing the accumulated rewards along an episode and maximizing it through reinforcement learning result in the optimal performance defined in Eq. 3.

E. The biological plausibility

The biological brain forms various cognitive functions by incorporating different sub-brain regions. In movement planning, it integrates the prefrontal cortex [39], hippocampus [40], and cerebellum [41]. The hippocampus has played key roles in long-term to short-term memory conversion by incorporating sequential movement planning in the prefrontal cortex as induction and spatially-temporal contact encoding in the cerebellum as decision making. The hippocampus can handle spatial information (e.g., place cell and grid cell for spatial position encoding [42]) and temporal information (e.g., sequential information storage and association [43]) towards efficient and robust cognitive computation. The prefrontal cortex is a six-layer brain region but with different subcortical columns for basic pattern recognition [44]. The cerebellum is also layer-wised but with fewer layers and more asymmetrical synapses [45], making it powerful for specialized sequential information processing. Here, we borrowed these ideas from the biological brain to build a motion-planning SNN, with more biologically plausible structures toward biologically efficient movement planning.

V. Teaching signal generation

In a supervised learning manner, we train the cerebellar network and coarsely learn the prefrontal cortex network. Therefore, we need to design an appropriate controller and planner and update the plastic efficacy to reach hand-crafted behaviors.

A. Compensational controller

In this paper, the cerebellar network aims at compensating for radial contacts. The radial force variation may be affected by the compensational movement, the variation due to primary movement, and the uncertainty. It is reasonable to consider both the current and predicted contacts to perform well in compensation. Another concern is uncertainty, which may come directly from unknown contact surfaces and indirectly from calibration errors. Therefore, a random radial force will emerge after each primary movement, even after compensation, which requires a controller that can handle random states for radial contact compensation. We describe a controller to avoid overshooting, whose radial movement takes the form

$$\boldsymbol{m}_{r,t} = \boldsymbol{p}_2 \circ e^{-(\tau - \boldsymbol{p}_1) \circ (\tau - \boldsymbol{p}_1)}, \tag{14}$$

where $\circ$ means the Hadamard product, $\boldsymbol{p}_1, \boldsymbol{p}_2 \in \Re^{2\times 1}$ are parameter vectors, and $\tau \in \Re^{2\times 1}$ is the time counted from the appearance of the force to be compensated. The form in Eq. 14 means that, for any given current and predicted radial forces, we expect the radial movement trajectory to be exponential to approximate zero without overshooting.

The parameters $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ are solved by differentiating the above equation and employing the elastic deformation model

$$\begin{cases} -\tilde{\boldsymbol{f}}_{r,t+1} = M\boldsymbol{p}_2 \circ e^{-\boldsymbol{p}_1 \circ \boldsymbol{p}_1}, \\ \dot{\boldsymbol{m}}_{r,t-1} = 2\boldsymbol{p}_2 \circ \boldsymbol{p}_1 \circ e^{-\boldsymbol{p}_1 \circ \boldsymbol{p}_1}, \end{cases} \tag{15}$$

where $\tilde{\boldsymbol{f}}_{r,t+1} = [f_{x,t} + \gamma_{r,x}\mu_{x,t+1}, f_{y,t} + \gamma_{r,y}\mu_{y,t+1}]^{\mathrm{T}}$ is the radial contact state considering the prediction values, M is the object stiffness acquired via experiment, and $\gamma_r \in (0,1]$ is the discounted parameter in the controller. In the above equation, the first row depicts the linear relationship between contact force and compensational movement; and the second row originates from the deviation of Eq. 14. In other words, Eqs. 14 and 15 give the expression of displacement and velocity. Note that

the parameter $\tilde{\boldsymbol{f}}_r$ is the sum of the current radial contact and the predicted values multiplying a discounted parameter. The parameter γ_r illustrates the confidence in the predicted contact

$$\gamma_r = \min\left\{\frac{\int_{-\sigma_0}^{\sigma_0}\mathcal{N}(0,\sigma_{t+1})\,df}{\int_{-\sigma_0}^{\sigma_0}\mathcal{N}(0,\sigma_0)\,df},1\right\}, \tag{16}$$

where σ_0 is the comparing standard variance. This equation obtains confidence as the ratio of the probability of the predicted distribution located between $\pm\sigma_0$ to the probability of the standard Gaussian distribution in the same range. σ_0 is similar to a threshold, and a standard variance less than or equal to it results in $\gamma_r = 1$, which means we can completely trust the predicted contact. The larger the standard variance, the lower the confidence of the prediction.

With the parameters, $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$, the acceleration yields

$$\ddot{\bar{\boldsymbol{m}}}_{r,t} = (4\boldsymbol{p}_2 \circ \boldsymbol{p}_1 \circ \boldsymbol{p}_1 - 2\boldsymbol{p}_2) \circ e^{-\boldsymbol{p}_1 \circ \boldsymbol{p}_1}, \tag{17}$$

where $\ddot{\bar{\boldsymbol{m}}}$ is the computed acceleration. This value saturates

$$\ddot{\boldsymbol{m}}_{r,t} = \begin{cases} \ddot{m}_{max}\frac{\ddot{\bar{\boldsymbol{m}}}_{r,t}}{\|\ddot{\bar{\boldsymbol{m}}}_{r,t}\|_2}, & \|\ddot{\bar{\boldsymbol{m}}}_{r,t}\|_2 - \ddot{m}_{max} > 0 \\ \ddot{\bar{\boldsymbol{m}}}_{r,t}, & \text{otherwise} \end{cases} \tag{18}$$

The outputs of the compensational controller result in

$$\dot{\boldsymbol{m}}_{r,t} = \ddot{\boldsymbol{m}}_{r,t}\Delta t + \dot{\boldsymbol{m}}_{r,t-1}, \qquad \bar{\boldsymbol{\mu}}_{t+1} = \boldsymbol{f}_{r,t} - M\dot{\boldsymbol{m}}_{r,t}\Delta t. \tag{19}$$

These provide teaching signals for the learning of the cerebellar network.

B. Movement planner

Given the inputs of the prefrontal cortex network, we first design a planner to generate axial movements considering previous confidence, current assessment, and future prediction.

We evaluate the current performance, based on radial contact and axial velocity

$$\mathscr{C}_{a,t} = 2e^{-\frac{\gamma_a\|f_{r,t}\|_2}{\|\dot{m}_{a,t-1}\|}} - 1, \tag{20}$$

where γ_a is a scaling parameter for the current assessment and $\mathscr{C}_{a,t} \in [0,1]$ is the defined assessment that monotonically increases with axial velocity and decreases with radial contact.

For the probability part, we expect a zero contact prediction and introduce a corresponding parameter

$$\mathscr{C}_{p,t} = \frac{1}{T_p}\int_{-\sigma_{x,t+n}}^{\sigma_{x,t+n}}\int_{-\sigma_{y,t+n}}^{\sigma_{y,t+n}}\mathcal{N}\left(\bar{\boldsymbol{f}}_{r,t+n},\boldsymbol{\Sigma}_{t+n}\right)d\boldsymbol{f}_r, \tag{21}$$

where $\mathscr{C}_{p,t}$ is the prediction parameter of n future steps, $\boldsymbol{\Sigma}_{t+n}$ is a matrix with $\sigma^2_{x,t+n}$ and $\sigma^2_{y,t+n}$ on its principal diagonal, and T_p means the probability of a standard multivariate Gaussian distribution located between $\pm\sigma_{x,t+n}$ and $\pm\sigma_{y,t+n}$ divided by the square of the prediction step

$$T_p = \int_{-\sqrt{n}\sigma_{x,t+1}}^{\sqrt{n}\sigma_{x,t+1}}\int_{-\sqrt{n}\sigma_{y,t+1}}^{\sqrt{n}\sigma_{y,t+1}}\mathcal{N}\left(\boldsymbol{0},n\boldsymbol{\Sigma}_{t+1}\right)d\boldsymbol{f}_r. \tag{22}$$

Similar to Eq. 16, the above two equations compute the future prediction parameter by considering the values that lie within standard variance.

We evaluate the confidence in prior performance

$$\beta_t = (1+\mathscr{C}_{a,t})\,\beta_{t-1}. \tag{23}$$

where $\beta_t \in [0,1]$ is an internal parameter, iteratively updating based on every step assessment.

Considering the above three parameters, we plan the axial velocity as

$$\bar{\dot{m}}_{a,t} = [1+\beta_t + (1+\mathscr{C}_{a,t})(\mathscr{C}_{p,t} - T_n)]\,\dot{m}_{a,t-1}, \tag{24}$$

where T_n is a positive parameter. The above equation presents a complex composition of using the three parameters to regulate the axial movement, and a simple example of quick understanding is that it expedites if $\beta_t + \mathscr{C}_{p,t} - T_n \geq 0$ holds. This leads to

$$\bar{\ddot{m}}_{a,t} = \frac{\min\{\bar{\dot{m}}_{a,t},\dot{m}_{max}\} - \dot{m}_{a,t-1}}{\Delta t}, \tag{25}$$

where $\bar{\ddot{m}}_a$ is the computed acceleration. An important signal indicating whether to decelerate is

$$s_l = \begin{cases} 1, & \min\{\bar{\ddot{m}}_{a,t},\ddot{m}_{max}\} \geq \sqrt{2\ddot{m}_{max}\left(l - \int_0^t \dot{m}_{a,\tau}d\tau\right)} \\ 0, & \text{otherwise} \end{cases} \tag{26}$$

where s_l is a variable indicating whether the desired length is close. This value saturates by its limit

$$\ddot{m}_{a,t} = \begin{cases} -\ddot{m}_{max}, & s_l = 1 \\ \bar{\ddot{m}}_{a,t}, & \|\bar{\ddot{m}}_{a,t}\| \leq \ddot{m}_{max} \\ \ddot{m}_{max}, & \text{otherwise} \end{cases} \tag{27}$$

The first row of this equation decelerates the movement when it is close to the desired length. The velocity output yields

$$\dot{m}_{a,t} = \ddot{m}_{a,t}\Delta t + \dot{m}_{a,t-1}. \tag{28}$$

In summary, we use $\mathscr{C}_{p,t}$ as the teaching signal for the function e_μ and s_l for the function e_l. $\dot{m}_{a,t}$ can be the teaching signal for the function e_p to coarsely tune the prefrontal cortex network.

VI. Experiments and Results

We verify the function of each module and test the whole structure in experiments. We set the parameters of neuron dynamics: the potential threshold $V_{th} = 0.5$ (a scaled version of the potential being 0 at rest and 1 at the maximum), the decay rate $k_r = 0.2$, and 50 ms for the training time window. We train our networks on NVIDIA TITAN Xp and learn the hippocampal module in about 10 hours with one GPU, the cerebellar module in an hour with eight GPUs, and the prefrontal cortex module in 48 hours with eight GPUs.

A. Module verification

To test the hippocampal network, we randomly select Gaussian distributions from experiments and learn the distribution parameters (two horizontal means and two variances) together with their temporal property. The red lines in Fig. 5 are the fitting results of the four distribution parameters for one prediction step, and the blue and green lines represent the

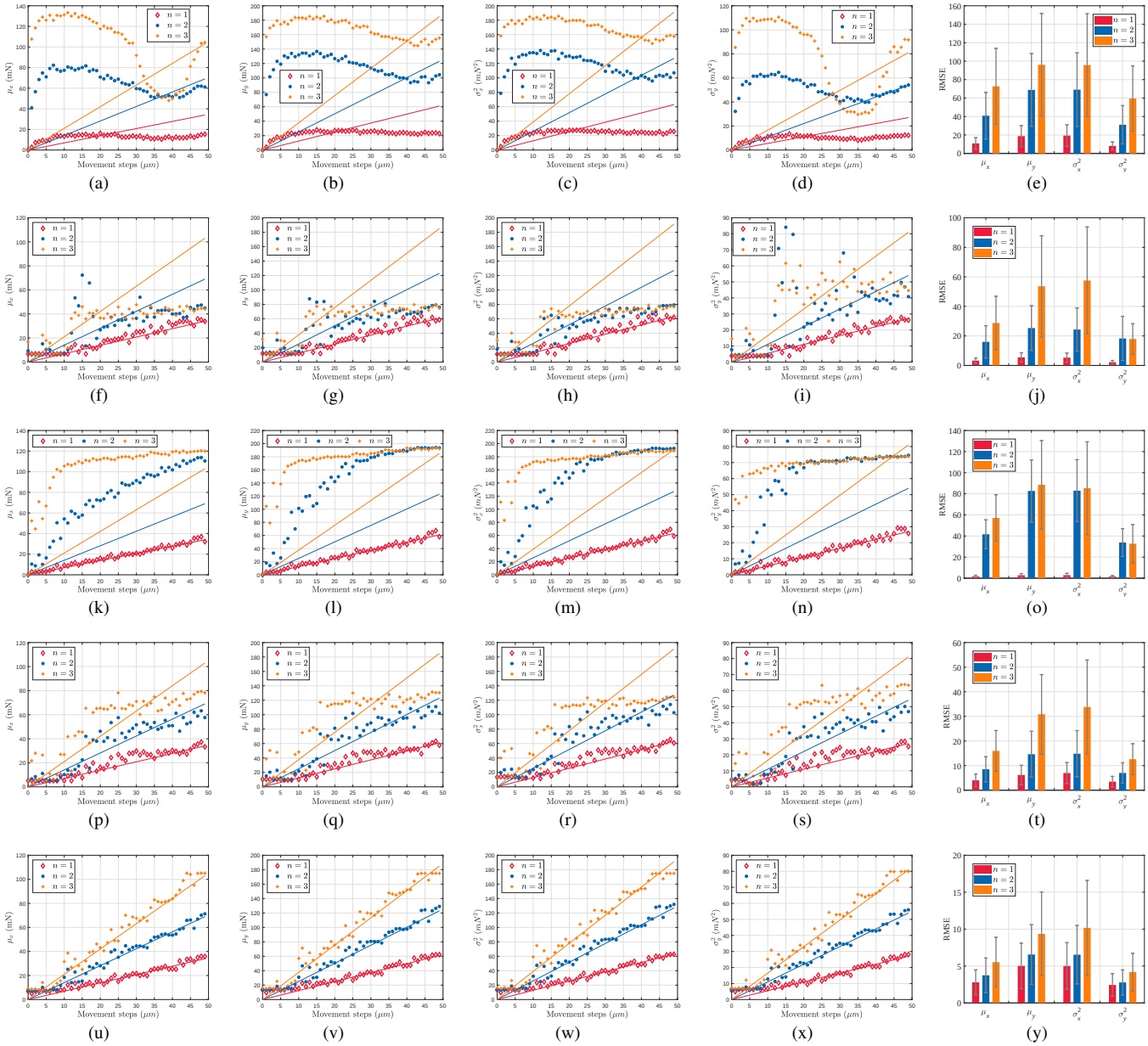
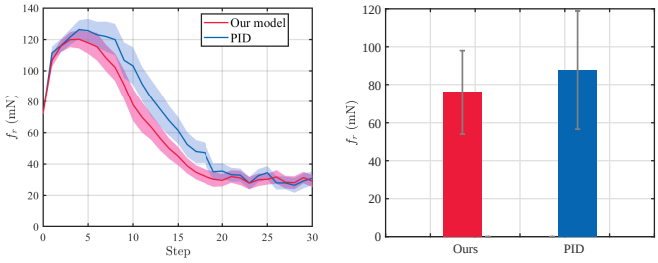


Fig. 5. The learning results of different networks imitating the hippocampus. The rows are the results of using feedforward network with feedback, LSM without feedback, RSNN without feedback, LSM with feedback, and RSNN with feedback, in sequence. The first four columns are the fitting errors of μ_x, μ_y, σ_x^2, and σ_y^2, and the last column is the corresponding error bars. The solid lines in the first four columns represent the desired parameters, and red, blue, and green corresponds to prediction steps of $n = 1, 2, 3$, respectively

prediction values for $n = 2$ and $n = 3$, respectively. These three lines conform to the temporal property inherent in Eq. 8.

To verify the effectiveness of the double recurrence, we compare the performance of several networks. All networks have 50 neurons in the input layer, 343 ($7 \times 7 \times 7$) neurons in the hidden layer, and 40 neurons in the output layer. The first network that we employ is a feedforward network with feedback, which represents an outer recurrence. We train it using the error function in Eq. 9, where $\boldsymbol{Q}_h$ is set as an identity matrix. A progressive training method is adopted, in which the n-step prediction network uses the weights of the $n-1$-step as initialization, and the learning rate is also progressively decreased. We present the learning results in the first row of Fig. 5. This shows that the network with only an outer recurrence has quite large errors in predicting the next step and several future steps. The prediction RMSE of four distribution parameters for $n = 1$ averages 14.3 with an averaged variance of 8.4. All fitting errors increase as the prediction goes further. After it, we employ two recurrent networks as examples to view the performance of inner recurrence. In other words, the



(a) (b)

Fig. 6. The compensational results of the cerebellar network. (a) The results of one hundred trajectories using our model and a PID controller. (b) The forces of the first 20 steps averaged over the 100 trajectories.

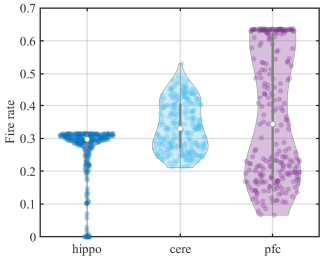


Fig. 7. The fire rates of each module, where the abbreviations of "hippo", "cere", and "pfc" represent the modules of the hippocampus, cerebellum, and prefrontal cortex.

training process only considers $n = 1$ while the testing process attempts $n = 2$ and $n = 3$. This manner is mainly determined by the structure where no feedback is supported. The second row in Fig. 5 shows the results of LSM without feedback, and the next row displays the results of the RSNN without feedback. Both networks generate fewer fitting errors for predicting one step, averaging 4.0 for LSM and 2.2 for RSNN. The comparison of the two networks shows the importance of trainable inner recurrence. This trainable-untrainable difference also affects the performance of predicting deeper into the future. The RMSE of LSM increases with the increment of prediction steps, and most predictions become increasingly smaller than the labeled values as the movement step increases. The RSNN behaves differently. Its outputs corresponding to small movement steps increase rapidly and then saturate, with similar RMSEs for $n = 2$ and $n = 3$. Comparing the first three rows in Fig. 5, we can conclude that the inclusion of inner recurrence can predict well for one step, and the inclusion of outer recurrence leads to a moderate error increment as the prediction steps increase.

Last, we use the two recurrent networks with feedback to verify the performance of double recurrence. The last two rows in Fig. 5 show the learning results. The LSM with feedback still presents an increasing trend as the prediction step n enlarges, averaged 5.2 for $n = 1$, 11.2 for $n = 2$, and 23.3 for $n = 3$. This fourfold average illustrates from another viewpoint that the network with untrainable inner recurrence cannot handle well the temporal property. However, with outer recurrence, it can feasibly reduce the prediction errors of the next several steps, 46% for $n = 2$ and 41% for $n = 3$. The first four subfigures of the fourth row in Fig. 5 show that the predictions are not well coincident with the labeled data, even for the next step, and their deviations become more severe as predicted further. That is the negative side of untrainable neurons. The RSNN with feedback has the best performance. Compared to the network without outer recurrence, it remarkably improves the accuracy of predicting deeper, at the expense of slightly increasing the prediction error for $n = 1$. Specifically, this double recurrent structure increases prediction errors from 2.2 to 3.8 for $n = 1$ but reduces from 60.3 (65.9) to 4.9 (7.3) for $n = 2(3)$. The first four subfigures of the last row in Fig. 5 reveal the coincidence of the predictions and the labeled data for each prediction step. This result yields the advantages of outer recurrence. Comparing the two double recurrent networks shows that these trainable neurons can deliberately increase the fitting accuracy for each prediction step. Viewing the first and last rows in Fig. 5, we conclude that the addition of inner recurrence improves the accuracy by 73% for $n = 1$ and 91% for the next two and three steps. All these results verify the effectiveness of the double recurrence in dealing with temporal property.

We test the cerebellar network and compare it with a PID controller. The network has 64 neurons in the MFs, 4000 in the GCs, 1600 in the PCs, and 8 in the DCN. We train it in a thousand iterations. The parameters of the comparative PID controller are regulated using the Ziegler-Nichols method. We randomly select the initial states: radial force $f_{r,0}$ in $[64, 80]$ mN and radial velocity $\dot{m}_{r,0}$ in $[-2, 0]$ $\mu m/s$. Suppose the velocity of the primary movement is 30 $\mu m/s$, and the parameters for contact prediction are shown in Fig. 5. In each step, we use Gaussian sampling to acquire contacts due to primary movements. Fig. 6(a) shows the results of one hundred trajectories. Since the initial radial velocity is negative, the forces increase first and then decrease gradually, and our model presents fewer radial forces than the PID controller since it considers the contact prediction. After the first 20 steps, the effect of large initial forces and negative velocities is compensated, and the controller mainly focuses on the contact caused by primary movements. That is why the forces do not approach zero since the next contact conforms to a distribution. Fig. 6(b) shows the mean force for each trajectory of the first 20 steps. Our model produces 76 mN radial forces on average, and the comparative controller generates 88 mN. These results show the advantage of the cerebellar network.

Compared with traditional networks, SNNs have the advantage of energy efficiency. We test each module on 256 iterations of computation and present their fire rates in Fig. 7. The firing rate varies with the inputs. Most neurons of the hippocampal module fire at a ratio of 0.3, and a smaller input corresponds to a lower fire rate. As the inputs are close to zero, all those neurons are at rest, as depicted in Fig. 5. The mean and the standard variance of the fire rate of the hippocampal module are 0.26 and 0.08, respectively. In contrast, the fire rate of the cerebellar module is relatively uniform, and the neurons fire at a ratio between 0.2 and 0.45 for different inputs. The mean and the standard variance of the fire rate of the cerebellar module are 0.34 and 0.08, respectively. The

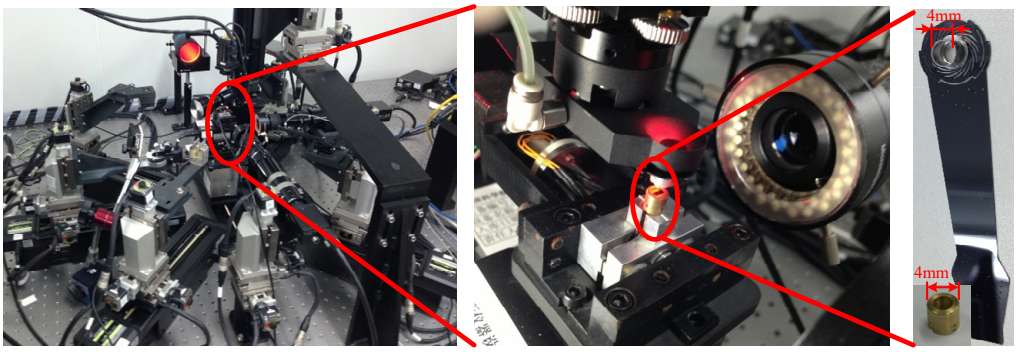
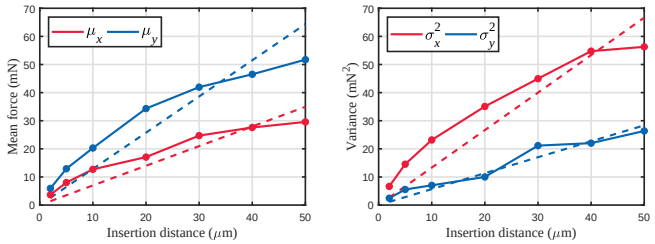


Fig. 8. The experimental platform to assemble objects in high precision.



(a) (b)

Fig. 9. Gaussian distribution verification and parameters acquired in experiments. (a) The mean force measured with insertion distance of 2 μm, 5 μm, 10 μm, 20 μm, 30 μm, 40 μm, and 50 μm. (b) The variance with the above feeding distances. The dotted lines are the fitting results to meet the requirement of Eq. 8.

TABLE I. THE PARAMETERS USED IN EXPERIMENTS.

$\dot{m}_{max}$	M	γ_a	γ_c	γ_f	T_n
100 $\mu m/s$	80 mN/μm	$0.5ln2$	1	1	0.5
$\ddot{m}_{max}$	l	σ_0	β_0	γ_h	n
20 $\mu m/s^2$	2.05 mm	0.1 $Newtons$	0.3	0.9	3

prefrontal cortex module has the largest fire rate range: Many neurons have fire rates as high as 0.63, but some are even lower than 0.1. The prefrontal cortex module fires with a mean of 0.37 and a standard variance of 0.20. The average result of the three modules is a fire ratio of 0.33, demonstrating energy conservation using spiking neurons.

B. Experiments

We test the proposed structure on a precision assembly platform, as shown in Fig. 8. The platform possesses three microscopes, two force sensors, and six robot arms in total. In this experiment, we employ only two microscopes, one force sensor, and two robot arms, which are sufficient for verification. The contact parts of the two objects are hollow cylinder-shaped: the hole is 50 μm thick and 4.1 mm in diameter; the peg is 50 μm thick, 4 mm long, and 4 mm in diameter. The interference between two objects lies in $0 \sim 5$ μm. The arm gripping the peg is equipped with three linear ball guides Suguar KWG06030-G, whose translational resolution is ± 0.5 μm, to achieve 3D translational movement and a two-axis rotation stage Sigma KKD-25C, for two-DOF manual tilt adjustment. The arm holding the hole has an elevator stage Micos ES-100, with movement errors within 0.1 μm, to fulfill the insertion and three motorized goniometer stages KGW06050-L, KGW06075-L, and SGSP-40yaw, for rotation. The force sensor is Nano-43 with a resolution of 1/128 $Newtons$, which can be improved with filtering. The horizontal microscopes are GC2450 and PointGrey 50S5M-C, used in the alignment process. Tab. I shows the parameters adopted in experiments. The two objects are 50 μm away after alignment, and the target is to insert the peg 2 mm into the hole, which leads to $l = 2.05$ mm. β_0 is the initial set of the inner parameter β_t in the prefrontal cortex. The parameter γ_a zeros the assessment $\mathscr{C}_{a,t}$ corresponding to maximum velocity and 200 mN radial contact.

We first verify whether the insertion conforms to a Gaussian distribution. We assemble the two objects to a randomly picked point, where we repeat to insert and withdraw with a fixed insertion step many times. Fig. 9 presents the mean forces and variances with insertion distances of 2, 5, 10, 20, 30, 40, and 50 μm. The mean radial force for insertion of 10 μm is $[12.7, 20.3]^{\mathrm{T}}$ mN, and the corresponding variance is $[23.2, 7.0]^{\mathrm{T}}$ mN^2. When the insertion step changes to 50 μm, the mean and variance become $[29.6, 51.7]^{\mathrm{T}}$ mN and $[56.3, 26.4]^{\mathrm{T}}$ mN^2. This shows that different insertions will result in different radial contacts. It also appears that the Gaussian parameters scale with the insertion step. The larger the insertion step, the larger the mean and variance of the radial contact. If the insertion velocity remains high, the increment of radial contacts due to insertion is not a negligible factor in consideration of compensation. The large variance of the high insertion velocity results in low confidence in the radial contact prediction. To meet the requirement of Eq. 8, we fit the curves with lines and use them to train the hippocampal model. To investigate the modeling errors, we present the proportions of the change in the radial force and the fitting curves with several insertion distances, as shown in Fig. 10. We organize the force changes into groups with an interval

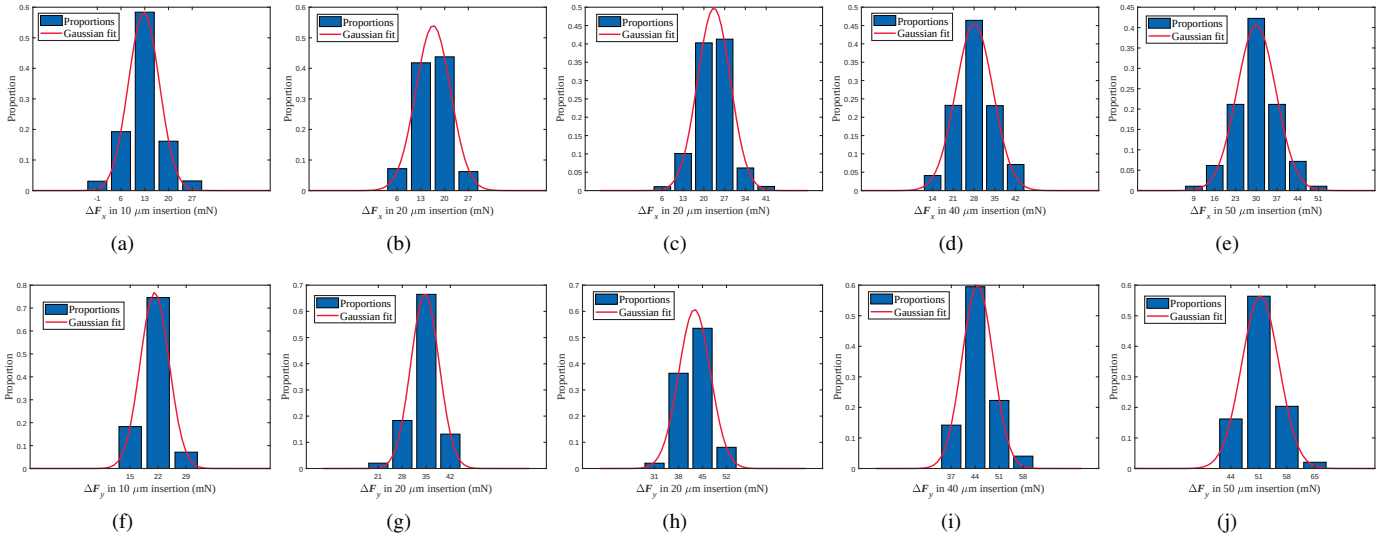


Fig. 10. Gaussian distribution of the radial forces measured with insertion distance of 10 μm, 20 μm, 30 μm, 40 μm, and 50 μm. The upper row lists the force in the x-axis, and the lower row lists the force in the y-axis.

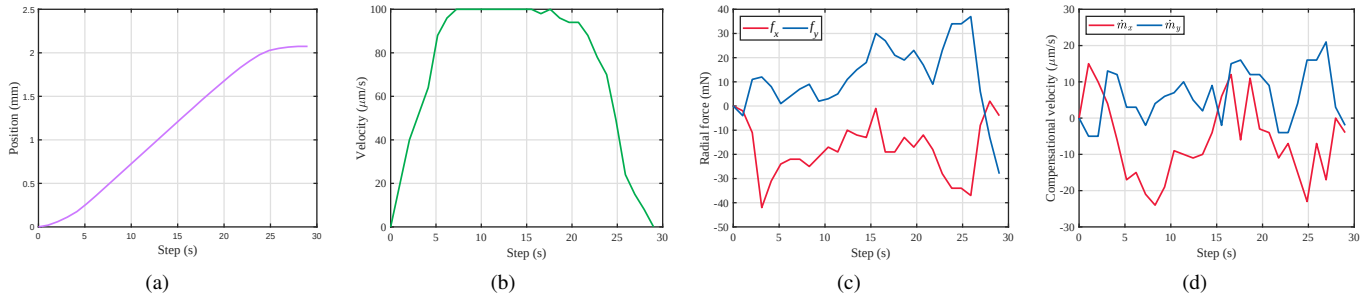
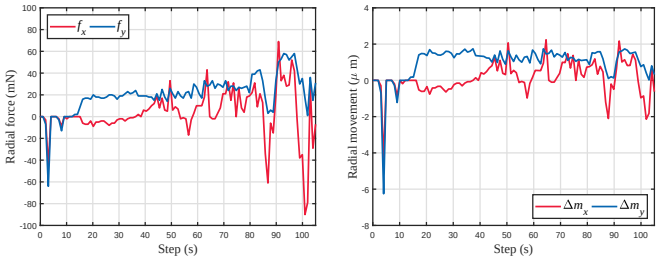


Fig. 11. The experiment results on precision assembly. (a) The total axial movement length. (b) The axial velocity $\dot{m}_{z,t}$. (c) The radial contact forces. (d) The velocity of the compensational movement $\dot{m}_{x,t}$ and $\dot{m}_{y,t}$.

of 7 mN, corresponding to the resolution of the force sensor. Viewing from left to right (increasing the insertion step), the fitted Gaussian curve shifts to the right (increasing its mean) and becomes flatter (increasing its standard variance). From top to bottom (comparing the two horizontal axes), the x-axis force has a smaller mean value but a larger standard deviation relative to the y-axis force. We measure the goodness of Gaussian fitting with the R-square, which are all beyond 0.96, experimentally verifying the correctness of using a Gaussian distribution to model the contact.

We employ the RSNN with feedback to learn this Gaussian distribution and apply the same network structure as used in the preceding subsection for the hippocampus and cerebellum. We select 64, 256, 512, 512, 128, and 4 neurons for each layer of the prefrontal cortex module and train this network. To verify the effectiveness of the proposed structure, we insert the peg into the hole and examine the performance in movement efficiency and radial compensation. Fig. 11 shows the inserting and contacting results. The inserting process consists of three parts: starting with maximum acceleration, steady inserting with full speed, and decelerating to the desired state, as shown in Fig. 11(b). The acceleration in the first and third parts is close to its maximum value, but it does not reach and maintain the maximum value like manually designed controllers. It takes 28 seconds to accomplish the task, as shown by the trajectory in Fig. 11(a). Figs. 11(c) and 11(d) illustrate the radial contact and velocity. The cerebellar module is trained to generate radial movements considering both the current force and the predicted contact based on the distribution parameters stored in the hippocampal module. The radial contact is limited to a small range while the insertion is fast, as shown in Figs. 11(b) and 11(c). Recall that for insertion of 50 μm, the radial force increases by approximately 59 mN. For a double insertion velocity, the radial contact in Fig. 11(c) is within 50 mN and averages 25 mN, taking into account the uncertainties and disturbances in insertion. These results show the effectiveness of the proposed structure.

We apply the force-based method proposed in [3] to carry out the insertion with the same objects and compare the experimental results. The comparison work uses an incremental PI controller, which is model-free, and the proportional and integral parameters are 0.5 and 0.3. Since this approach has no capability of predicting future contact, we set a fixed insertion step $\Delta m_z = 20\ \mu m$, a conservative insertion strategy, to guaran-



(a) (b)

Fig. 12. The compared experiment results. (a) The radial contact force. (b) The radial movement.

tee acceptable interaction forces after each insertion. A slightly deeper insertion than this picked step will inevitably produce much larger contact forces. We apply the same initialization and target, and Fig. 12 shows the experimental results. As the peg inserts deep, the radial forces increase, and the largest contact is close to 100 *mN*. Viewing Figs. 12(a) and 11(c), we note that the proposed approach spends less than one-third of the steps while producing half of the maximum radial contact compared with the method in [3]. This means that even with a conservative insertion strategy, the comparative controller still generates larger contacts. This comparison shows the advantage of the proposed approach.

We carry out another comparison experiment, in which a model-based method in [7] is employed to control insertion based on contact prediction. The comparison uses a Gaussian distribution to fit the radial force variation and, based on it, plans the axial movement. It does not consider the predicted force in the radial compensation. Fig. 13 shows the performance of applying this comparative method. This manually designed controller takes the maximum acceleration in the accelerating and decelerating parts, ending in 26 steps, but leads to larger radial forces. As shown in Fig. 13(c), the radial force reaches 160 *mN* at some steps and averages 78 *mN*. Compared with it, the proposed method produces only one-third of radial forces. These results demonstrate the advantage of the proposed method from another viewpoint.

VII. Conclusions

This paper presents a novel structure to imitate the joint function of multiple brain regions for precision physical interaction. We use three modules to simulate the memory function of the hippocampus, the controlling function of the cerebellum, and the planning function of the prefrontal cortex. In the hippocampus, we propose a double recurrent network to consider the spatiotemporal property of the Gaussian distribution. We apply a cerebellar module to compensate for radial forces. Uniting these two brain regions, we can predict the radial contact of a series of future steps after considering the compensation capability. We simulate the prefrontal cortex to plan axial movements based on past performance, current assessment, and future prediction. The cooperation of these three modules mimics the joint function of corresponding brain regions. We provide an appropriate controller and planner for teaching signal generation and use reinforcement learning to fine-tune the network. Experiments validate the proposed method.

In the future, we will extend this research to more complicated scenarios in robot planning and control, mimicking joint functions of more brain regions. We will also investigate how to leverage these studies in robots to explain phenomena in neuronal activity.

Appendix A
The Temporal Property of the Movements Conforming to Gaussian Distributions

In this part, we show how to predict n successive steps and demonstrate the temporal property. We expand Eq. 1 in the following form

$$P\left(\boldsymbol{f}_{r,t+1}|\boldsymbol{f}_{r,t},\int_t^{t+1}\dot{m}_a dt\right)=\mathcal{N}\left(\boldsymbol{f}_{r,t+1}|\boldsymbol{f}_{r,t}+\boldsymbol{\mu}_{\dot{m}_{a,t}},\boldsymbol{\Sigma}_{\dot{m}_{a,t}}\right). \tag{29}$$

Its difference from Eq. 2 is that we replace the subscript of distribution parameters from time-related to movement-related to facilitate later formula derivation. Based on the above equation, we can acquire the probability of the contact at the time $t+2$

$$\begin{aligned}&P\left(\boldsymbol{f}_{r,t+2}|\boldsymbol{f}_{r,t+1},\int_{t+1}^{t+2}\dot{m}_a dt\right)\\&=\int\mathcal{N}\left(\boldsymbol{f}_{r,t+2}|\boldsymbol{f}_{r,t+1}+\boldsymbol{\mu}_{\dot{m}_{a,t+1}},\boldsymbol{\Sigma}_{\dot{m}_{a,t+1}}\right)\\&\mathcal{N}\left(\boldsymbol{f}_{r,t+1}|\boldsymbol{f}_{r,t}+\boldsymbol{\mu}_{\dot{m}_{a,t}},\boldsymbol{\Sigma}_{\dot{m}_{a,t}}\right)d\boldsymbol{f}_{r,t+1}.\end{aligned} \tag{30}$$

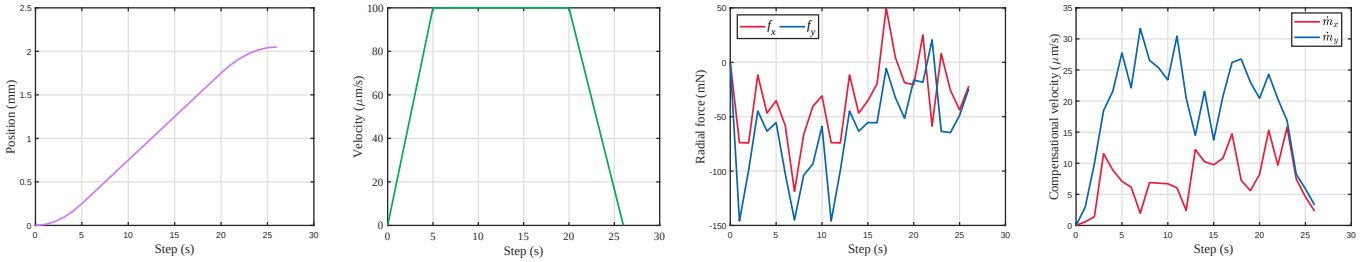
The above equation integrates all the possibilities that the movement from $t+1$ to $t+2$ transits from all possible $\boldsymbol{f}_{r,t+1}$ to $\boldsymbol{f}_{r,t+2}$. Because the mean value of the distribution of $\boldsymbol{f}_{r,t+2}$ is linearly dependent on $\boldsymbol{f}_{r,t+1}$, we can translate $\boldsymbol{f}_{r,t+2}$ by subtracting $\boldsymbol{f}_{r,t+1}$, which yields

$$\begin{aligned}&P\left(\boldsymbol{f}_{r,t+2}|\boldsymbol{f}_{r,t+1},\int_{t+1}^{t+2}\dot{m}_a dt\right)\\&=\int\mathcal{N}\left(\boldsymbol{f}_{r,t+2}-\boldsymbol{f}_{r,t+1}|\boldsymbol{\mu}_{\dot{m}_{a,t+1}},\boldsymbol{\Sigma}_{\dot{m}_{a,t+1}}\right)\\&\mathcal{N}\left(\boldsymbol{f}_{r,t+1}|\boldsymbol{f}_{r,t}+\boldsymbol{\mu}_{\dot{m}_{a,t}},\boldsymbol{\Sigma}_{\dot{m}_{a,t}}\right)d\boldsymbol{f}_{r,t+1}.\end{aligned} \tag{31}$$

According to the definition of convolution, we can write the probability of $\boldsymbol{f}_{r,t+2}$ in a convolution form of two Gaussian distributions

$$\begin{aligned}&P\left(\boldsymbol{f}_{r,t+2}|\boldsymbol{f}_{r,t+1},\int_{t+1}^{t+2}\dot{m}_a dt\right)\\&=\mathcal{N}\left(\boldsymbol{f}_{r,t+2}-\boldsymbol{f}_{r,t+1}|\boldsymbol{\mu}_{\dot{m}_{a,t+1}},\boldsymbol{\Sigma}_{\dot{m}_{a,t+1}}\right)\\&*\mathcal{N}\left(\boldsymbol{f}_{r,t+1}|\boldsymbol{f}_{r,t}+\boldsymbol{\mu}_{\dot{m}_{a,t}},\boldsymbol{\Sigma}_{\dot{m}_{a,t}}\right).\end{aligned} \tag{32}$$

The next primary movement $\dot{m}_{a,t+1}$ can be arbitrary, but since we aim to predict future contacts based on the current planning, it is reasonable to assume that the primary movement holds its value, i.e., $\boldsymbol{\mu}_{\dot{m}_{a,t+1}}=\boldsymbol{\mu}_{\dot{m}_{a,t}}$ and $\boldsymbol{\Sigma}_{\dot{m}_{a,t+1}}=\boldsymbol{\Sigma}_{\dot{m}_{a,t}}$. With this



(a) (b) (c) (d)

Fig. 13. The results of the second comparative experiment. (a) The total axial movement length. (b) The axial velocity $\dot{m}_{z,t}$. (c) The radial contact forces. (d) The velocity of the reaction movement $\dot{m}_{x,t}$ and $\dot{m}_{y,t}$.

assumption, we apply the Fourier transform and inverse Fourier transform to the above equation and obtain the final result of the probability of $\boldsymbol{f}_{r,t+2}$

$$P\left(\boldsymbol{f}_{r,t+2}|\boldsymbol{f}_{r,t+1},\int_{t+1}^{t+2}\dot{m}_a dt\right)=\mathcal{N}\left(\boldsymbol{f}_{r,t+2}|\boldsymbol{f}_{r,t}+2\boldsymbol{\mu}_{\dot{m}_{a,t}},2\boldsymbol{\Sigma}_{\dot{m}_{a,t}}\right). \tag{33}$$

We repeat this process to acquire the distribution of future n steps and similar results are acquired

$$\begin{aligned} &P\left(\boldsymbol{f}_{r,t+n}|\boldsymbol{f}_{r,t+n-1},\int_{t+n-1}^{t+n}\dot{m}_a dt\right)\\ &=\mathcal{N}\left(\boldsymbol{f}_{r,t+n}|\boldsymbol{f}_{r,t}+n\boldsymbol{\mu}_{\dot{m}_{a,t}},n\boldsymbol{\Sigma}_{\dot{m}_{a,t}}\right). \end{aligned} \tag{34}$$

From the above equation, we conclude that

$$\boldsymbol{\mu}_{t+n}=n\boldsymbol{\mu}_t,\quad \boldsymbol{\Sigma}_{t+n}=n\boldsymbol{\Sigma}_t. \tag{35}$$

This equation is the general expansion of Eq. 8, where constant movement velocity is assumed for the considered future steps. The linear property is based on the temporal sequence and should be applied to most of the movement. This leads to another relationship: the distribution parameters are the same for the double movement velocity in one time-step and one velocity in double time-step, i.e., $\boldsymbol{\mu}_{\dot{m}_{a,t+2}}=\boldsymbol{\mu}_{2\dot{m}_{a,t+1}}$ and $\boldsymbol{\Sigma}_{\dot{m}_{a,t+2}}=\boldsymbol{\Sigma}_{2\dot{m}_{a,t+1}}$. In other words, the parameters versus movement velocities are lines, and the difference between the lines for n time-steps and once step mainly exists in the slope, as shown in Fig. 5.

REFERENCES

[1] G. Lentini, A. Settimi, D. Caporale, *et al*., "Alter-Ego: A mobile robot with a functionally anthropomorphic upper body designed for physical interaction", *IEEE Robot. Autom. Mag.*, vol. 26, pp. 4, pp. 94-107, 2019.

[2] G. Xiang and J. Su, "Task-Oriented deep reinforcement learning for robotic skill acquisition and control", *IEEE Trans. Cybern.*, vol. 51, no. 2, pp. 1056-1069, 2021.

[3] D. Xing, D. Xu, F. Liu, *et al*., "Precision assembly among multiple thin objects with various fit types", *IEEE/ASME Trans. Mechatron.*, vol. 21, no. 1, pp. 364-378, 2016.

[4] A. Fakhari, M. Keshmiri, I. Kao, *et al*., "Slippage control in soft finger grasping and manipulation", *Adv. Robot.*, vol. 30, no. 2, pp. 97-108, 2016.

[5] F. Drigalski, L. E. Hafi, P. M. Eljuri, *et al*., "Vibration-Reducing end effector for automation of drilling tasks in aircraft manufacturing", *IEEE Robot. Autom. Lett.*, vol. 2, no. 4, pp. 2316-2321, 2017.

[6] D. Xing, F. Liu, F. Qin, *et al*., "Coordinated insertion control for inclined precision assembly", *IEEE Trans. Ind. Electron.*, vol. 63, no. 5, pp. 2990-2999, 2016.

[7] D. Xing, F. Liu, and D. Xu, "Efficient coordinated control strategy to handle randomized inclination in precision assembly", *IEEE Trans. Ind. Inform.*, vol. 16, no. 9, pp. 5814-5824, 2020.

[8] Y. Golan, A. Shapiro, and E. Rimon, "Jamming-Free immobilizing grasps using dual-friction robotic fingertips", *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 2889-2896, 2020.

[9] C. Yang, G. Peng, Y. Li, *et al*., "Neural networks enhanced adaptive admittance control of optimized robot-environment interaction", *IEEE Trans. Cybern.*, vol. 49, no. 7, pp. 2568-2579, 2019.

[10] W. Maass, "Networks of spiking neurons: The third generation of neural network models", *Neural Netw.* vol. 10, no. 9, pp. 1659-1671, 1997.

[11] S. Guo, Z. Yu, F. Deng, *et al*., "Hierarchical bayesian inference and learning in spiking neural networks", *IEEE Trans. Cybern.*, vol. 49, no. 1, pp. 133-145, 2019.

[12] S. Ghosh-Dastidar and H. Adeli, "Spiking neural networks", *Int. J. Neural Syst.*, vol. 19, no. 4, pp. 295-308, 2009.

[13] W. Maass, "Searching for principles of brain computation", *Curr. Opin. Behav. Sci.*, vol. 11, pp. 81-92, 2016.

[14] K. Roy, J. Akhilesh, and P. Priyadarshini, "Towards spike-based machine intelligence with neuromorphic computing", *Nature*, vol. 575, no. 7784, pp. 607-617, 2019.

[15] Z. Bing, C. Meschede, F. Rohrbein, *et al*., "A survey of robotics control based on learning-inspired spiking neural networks", *Front. Neurorobotics*, vol. 12, pp. 35:1-22, 2018.

[16] E. Nichols, L. J. McDaid, and N. Siddique, "Biologically inspired SNN for robot control", *IEEE Trans. Cybern.*, vol. 43, no. 1, pp. 115-128, 2013.

[17] A. Antonietti, D. Martina, C. Casellato, *et al*., "Control of a humanoid NAO robot by an adaptive bioinspired cerebellar module in 3D motion tasks", *Comput. Intell. Neurosci.*, vol. 2019, pp. 4862157:1-15, 2019.

[18] J. D. Wason, J. T. Wen, J. J. Gorman, *et al*., "Automated multiprobe microassembly using vision feedback", *IEEE Trans. Robot.*, vol. 28, no. 5, pp. 1090-1103, 2012.

[19] S. Huang and J. M. Schimmels, "Admittance selection conditions for frictionless force-guided assembly of polyhedral parts in two single-point principal contacts", *IEEE Trans. Robot.*, vol. 24, no. 2, pp. 461-468, 2008.

[20] T. Tang, H. Lin, Y. Zhao, *et al*., "Teach industrial robots peg-hole-insertion by human demonstration", *IEEE Int. Conf. Adv. Intell. Mechatron.*, pp. 488-494, 2016.

[21] S. Liu, D. Xu, D. Zhang, *et al*., "High precision automatic assembly based onmicroscopic vision and force information", *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 1, pp. 382-393, 2016.

[22] M. Lechner, R. Hasani, A. Amini, *et al*., "Neural circuit policies

enabling auditable autonomy", *Nat. Mach. Intell.*, vol. 2, no. 10, pp. 642-652, 2020.

[23] J. Wolfe, A. R. Houweling, and M. Brecht, "Sparse and powerful cortical spikes", *Curr. Opin. Neurobiol.*, vol. 20, no. 3, pp. 306-312, 2010.

[24] A. Sengupta, Y. Ye, R. Wang, *et al*., "Going deeper in spiking neural networks: VGG and residual architectures", *Front. Neurosci.*, vol. 13, pp. 95:1-10, 2019.

[25] B. Rueckauer, I. Lungu, Y. Hu, *et al*., "Conversion of continuous-valued deep networks to efficient event-driven networks for image classification", *Front. Neurosci.*, vol. 11, pp. 682:1-12, 2017.

[26] M. Pfeiffer and T. Pfeil, "Deep learning with spiking neurons: Opportunities and challenges", *Front. Neurosci*, vol. 12, pp. 774:1-18, 2018.

[27] G. Tang, A. Shah, K. P. Michmizos, *et al*., "Spiking neural network on neuromorphic hardware for energy-efficient unidimensional SLAM", *IEEE/RSJ Int. Conf. Intell. Rob. Syst.*, pp. 4176-4181, 2019.

[28] T. Hwu, A. Y. Wang, N. Oros, *et al*., "Adaptive robot path planning using a spiking neuron algorithm with axonal delays", *IEEE Trans. Cogn. Dev. Syst.*, vol. 10, no. 2, pp. 126-137, 2018.

[29] I. Abadfa, F. Naveros, J. A. Garrido, *et al*., "On robot compliance: A cerebellum control approach", *IEEE Trans. Cybern.*, vol. 51, no. 5, pp. 2476-2489, 2021.

[30] A. Gilra and W. Gerstner, "Predicting non-linear dynamics by stable local learning in a recurrent spiking neural network", *eLife*, vol. 6, pp. e28295:1-37, 2017.

[31] J. C. Tieck, S. Weber, T. C. Stewart, *et al*., "A spiking network classifies human sEMG signals and triggers finger reflexes on a robotic hand", *Robot. Auton. Syst.*, vol. 131, pp. 103566:1-10, 2020.

[32] I. Abadia, F. Naveros, E. Ros, *et al*., "A cerebellar-based solution to the nondeterministic time delay problem in robotic control", *Sci. Robot.*, vol. 6, no. 58, pp. eabf2756, 2021.

[33] C. Eliasmith, T. C. Stewart, X. Choo, *et al*., "A large-scale model of the functioning brain", *science*, vol. 338. no. 6111, pp. 1202-1205, 2012.

[34] W. Nicola and C. Clopath, "Supervised learning in spiking neural networks with FORCE training", *Nat. Commun.*, vol. 8, pp. 2208:1-15, 2017.

[35] Y. Wu, L. Deng, G. Li, *et al*., "Spatio-Temporal backpropagation for training high-performance spiking neural networks", *Front. Neurosci.*, vol. 12, pp. 331:1-12, 2018.

[36] W. Maass, T. Natschlaeger, and H. Markram, "Real-Time computing without stable states: A new framework for neural computation based on perturbations", *Neural Comput.*, vol. 14, no. 11, pp. 2531-2560, 2002.

[37] S. Wozniak, A. Pantazi, T. Bohnstingl, *et al*., "Deep learning incorporating biologically inspired neural dynamics and in-memory computing", *Nat. Mach. Intell.*, vol. 2, pp. 325-336, 2020.

[38] M. Ito, "The cerebellum: Brain for an implicit self", Pearson, 2011.

[39] J. Tanji, K. Shima, and H. Mushiake, "Concept-Based behavioral planning and the lateral prefrontal cortex", *Trends Cogn. Sci.*, vol. 11, no. 12, pp. 528-534, 2007.

[40] M. P. Witter, "Connectivity of the hippocampus", Springer New York, 2010.

[41] L. Casartelli, A. Federici, A. Cesareo, *et al*., "Role of the cerebellum in high stages of motor planning hierarchy", *J. Neurophysiol.*, vol. 117, no. 4, pp. 1474-1482, 2017.

[42] E. I. Moser, E. Kropff, and M. B. Moser, "Place cells, grid cells, and the brain's spatial representation system", *Annu. Rev. Neurosci.*, vol. 31, pp. 69-89, 2008.

[43] V. Cutsuridis, S. Cobb, and B. P. Graham, "Encoding and retrieval in a model of the hippocampal CA1 microcircuit", *Hippocampus*, vol. 20, no. 3, pp. 423-446, 2009.

[44] J. M. Fuster, "The prefrontal cortex - An update: Time is of the essence", *Neuron*, vol. 30, no. 2, pp. 319-333, 2001.

[45] J. DeFelipe, P. Marco, I. Busturia, *et al*., "Estimation of the number of synapses in the cerebral cortex: methodological considerations", *Cereb. Cortex*, vol. 9, no. 7, pp. 722-732, 1999.

Dengpeng Xing received the B.S. degree in mechanical electronics and the M.S. degree in mechanical manufacturing and automation from Tianjin University, Tianjin, China, in 2002 and 2006, and the Ph.D. degree in control science and engineering from Shanghai Jiao Tong University, Shanghai, China, in 2010.

He is currently an Associate Professor in the Research Center for Brain-Inspired Intelligence, Institute of Automation, Chinese Academy of Sciences. His research interests include brain-inspired robotics, robot control and learning.

Yiming Yang received the B.E. degree in electronic engineering from Beijing Institute of Technology, China in 2019. He is currently a PhD candidate in the School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China, and also with Research Center for Brain-Inspired Intelligence, Institute of Automation, Chinese Academy of Sciences, Beijing.

His research interests include spiking neural networks and reinforcement learning.

Tielin Zhang received the B.S. and M.S. degrees in telecommunication from Beijing University of Technology, Beijing, China, in 2010 and 2013, respectively, and received the Ph.D degree from the Institute of Automation Chinese Academy of Sciences, Beijing, China, in 2016.

He is an Associate Professor in the Research Center for Brain-Inspired Intelligence, Institute of Automation, Chinese Academy of Sciences, Beijing, China. His current interests include theoretical research on neural dynamics, Spiking Neural Networks, and neuromorphic chips.

Bo Xu received the B.S. degree in electrical engineering from Zhejiang University in 1988, and the M.S. and Ph.D. Degrees in pattern recognition and intelligent system from the Institute of Automation, Chinese Academy of Sciences in 1992 and 1997, respectively.

He is currently director of the Institute of Automation, Chinese Academy of Sciences. His research interests cover computational auditory model, spoken dialogue interaction, brain-inspired machine cognition, and decision making intelligence for complex system.