A Neurorobotic Embodiment for Exploring the Dynamical Interactions of a Spiking Cerebellar Model and a Robot Arm During Vision-Based Manipulation Tasks

Omar Zahra*, David Navarro-Alarcon* and Silvia Tolu†,‡
*The Hong Kong Polytechnic University, Kowloon, Hong Kong
†Technical University of Denmark, Kongens Lyngby, Denmark
‡stolu@elektro.dtu.dk

Received 13 April 2021
Accepted 14 April 2021
Published Online 18 May 2021

While the original goal for developing robots is replacing humans in dangerous and tedious tasks, the final target shall be completely mimicking the human cognitive and motor behavior. Hence, building detailed computational models for the human brain is one of the reasonable ways to attain this. The cerebellum is one of the key players in our neural system to guarantee dexterous manipulation and coordinated movements as concluded from lesions in that region. Studies suggest that it acts as a forward model providing anticipatory corrections for the sensory signals based on observed discrepancies from the reference values. While most studies consider providing the teaching signal as error in joint-space, few studies consider the error in task-space and even fewer consider the spiking nature of the cerebellum on the cellular-level. In this study, a detailed cellular-level forward cerebellar model is developed, including modeling of Golgi and Basket cells which are usually neglected in previous studies. To preserve the biological features of the cerebellum in the developed model, a hyperparameter optimization method tunes the network accordingly. The efficiency and biological plausibility of the proposed cerebellar-based controller is then demonstrated under different robotic manipulation tasks reproducing motor behavior observed in human reaching experiments.

Keywords: Spiking neural networks; cerebellum; robotic manipulation; sensor-based control; cellular-level; spiking.

1. Introduction

Neurorobotics is gaining more attention nowadays not only for improving the performance of robot controllers but also to reveal some of the mysteries about how our brains work. Limitation of state-of-the-art techniques to monitor spiking activity of all neurons in the brain makes it necessary to develop accurate computational models to verify theories about neural brain mechanisms. Also, the mutual benefit that derives from the joint research in neuroscience and robotics fields enables the development of adaptive biologically inspired controllers that can be used as a basis to explain mechanisms of learning and enhance the performance of robots. Hence, the motor system lies among the most studied for the common interest in both fields. Each of the brain regions contributing to the motor control has distinctive features leading to different roles in the control process. This study is concerned with modeling an essential complex region in our motor system, the cerebellum. The cerebellum is well known to help achieve fine motor control and precise timing and coordination of the movements of the joints to achieve dexterous motion. That was proven by studies of patients with lesions in the cerebellum suffering from clumsy
The aim of this work is to develop a controller based on a cerebellar-like model, developed on the cellular-level, to guide the motion of robots with real-time sensory information. The cerebellar model developed in this study is more detailed from the biological perspective to the previously developed models. The controller first builds a sensorimotor differential map through motor babbling and the cerebellum acts as a Smith predictor to correct discrepancies in sensory readings to enhance accuracy and precision of robot movements. Most of the previous studies included a forward and inverse cerebellar model to act together. In their case, the forward model provides sensory predictions while the inverse model provides corrections to the motor commands. Such hybrid approach is well appreciated from the control perspective but it has less biological evidence compared to the forward cerebellar model.

While developing cellular-level model adds more challenges for tuning and constructing the network, it provides insight into the real working of the biological counterparts and allows to import additional features from the wide repository of studies exploiting neural mechanisms to achieve such features. Building a detailed cellular level model requires utilizing spiking neural network (SNN), the third generation of artificial neural networks (ANN), to give a more faithful representation of the neural dynamics which provide them with more complex and realistic firing patterns based on spikes. The SNN adds a temporal dimension compared to the previous generations of ANN, which allows for developing more biologically realistic learning mechanisms and a more efficient representation of the spaces/dimensions encoded.

In Ref. [13], a controller is developed based on an inverse cellular-level cerebellar model to enhance the robot movement. The controller relies on a trajectory planner and inverse kinematics model to generate reference signals for angular position and speed. This limits the ability to learn only the manipulation of the robot based on the given reference value and is not suitable to learn directly from sensory feedback. In Ref. [23], a cerebellar model based on the adaptive filter theory is developed combining some computational neuroscience techniques along with machine learning. Such combination aims to achieve the sensorimotor adaptation but it does not model the spiking activity in the cerebellum. In Ref. [17], a cellular level cerebellar model is developed in which the teaching signal is provided based on the error in task-space. In all the previous studies, no clear method was identified to tune the network parameters, and the developed networks were used to manipulate only the end-effector in some predefined trajectory or to reach a certain target.

This study contributes to building a detailed spiking cellular-level cerebellar model including more biological features compared to the previous studies. The parameters of the network are set using a Bayesian optimization method to preserve several biological features observed in the cerebellum. This is demonstrated by monitoring the activity and firing rates of the different groups of neurons in the cerebellum, and the output from each layer, which is then compared to those obtained from biological counterparts. The teaching signal is provided as the error in task-space (TS) and demonstrates the ability to adapt to executing different tasks and handling manipulation of deformable objects. Additionally, it shall be noted that interaction with the environment plays a critical role in the emergence of various features in our nervous system and in properly validating the proposed models in comparison with the biological counterparts. Hence, the developed model is a fine compromise between the biological plausibility and accurate predictions of robot states. In future studies, the developed model will be utilized to study lesions in the cerebellum.

In the next sections, the preliminaries are introduced (Sec. 2), then modeling of the cerebellar controller and the hyperparameter optimization method.
A Neurorobotic Embodiment for Exploring the Dynamical Interactions of a Spiking Cerebellar Model

is described (Sec. 3). The experimental setup along with the results are explored (Sec. 4). Finally, an analysis of the obtained results is discussed and the main conclusions are derived (Sec. 5).

2. Preliminaries

The motor system is composed of several regions, each of which is responsible for a different function. These areas follow a hierarchical organization, as shown in Fig. 1(a). In such hierarchy, both parallel and serial connections provide different behaviors in different situations (i.e. dependent upon sensory feedback).

The higher order areas are responsible for decision making and planning the sequence of motion while coordinating the activity of several limbs. Lower order areas, on the other hand, control muscles while regulating forces and velocities with changes in posture and various interactions with the environment. The higher level tasks start from the cortical association areas and prefrontal cortex (along with the Basal Ganglia), which receive sensory information (from the sensory cortex) to generate an abstract plan for motion and the sequence of execution. This plan is then transformed to motor commands in the motor cortex which send these commands to the brain stem and the cerebellum (as an efference copy). The cerebellum plays a role in the coordination of movements and adjustment of timing to attain fine movements. Signals from the motor cortex travel to trigger motor neurons which innervate the skeletal muscles. The motor neurons in the spinal cord control the limbs and the movement of the body, while those in the brain stem control facial and head movements. In relation with the critical role of the cerebellum in both planning and execution of motion, this study focuses on the cerebellar corrections due to noisy sensory readings to obtain fine movements while executing the motor commands generated by the motor cortex to reach a target point in the space. Thus, a computational model of the cerebellum can help improve the performance of robotic controllers. In this work, a biologically inspired control system is built to guide a robot in a serving task after performing motor babbling for several iterations, as shown in Fig. 1. Computational spiking models are developed to both form a coarse sensorimotor map through motor babbling, and to reproduce the cerebellum. The sensory input to the formed map is modulated through the cerebellum to enhance the movement and reduce the deviation from the desired path. The developed cerebellar controller is capable of guiding the robot based on real-time noisy data.

3. Methods

3.1. The cerebellar-based control architecture

As discussed earlier, various theories were developed about the formation of the cerebellar forward or inverse models. However, more biological evidences support the theory that the cerebellum acts as a forward model based on the behavior observed in the case of cerebellar damage. In this study, the cerebellum acts as a computational forward model to predict the next sensory states based on the desired spatial velocity and the current robot states. The developed model predicts the robot state in TS which differs from other studies where the cerebellar forward model provides predictions in joint-space (JS). Predictions in JS provide a direct reference value to correct the angular positions and velocities, however, a reference value for optimal sensory signals from
joints shall be provided through an inverse model. On the other hand, predictions in TS allow to directly learn from errors related to the executed task without the need for an inverse model which makes it more versatile to learn novel tasks and skills. This model fits in the designed controller to act as a Smith predictor which is known to be capable of handling control schemes with long dead time as shown in Fig. 1. Analogous to control systems, in which dead-time is introduced due to the time needed for sensing, processing of the inputs, computing the control output and actuation, in biological systems the dead-time is introduced due to the time needed for the sensory signals to travel through the nervous system, generate a motor command and travel back for execution to the muscles. In our system, the dead time (expressed as a delay (D)) is caused by slow sensory readings (S) and robot (R) action. In this context, the motor cortex acts as a DM that can generate commands (u) in the JS based on the desired spatial velocity \( v^* \) and current joint angles \( q \), and thus acts as an inverse model, as shown in Fig. 2.

This can be expressed as

\[
u(t) = f(q(t), v^*(t)), \tag{1}
\]

where \( f \) represents the inverse mapping formed in the motor cortex to correlate the JS and TS. In case the robot moves from the current end-effector position \( x \) to a target position \( x^* \), \( v^* \) can be formulated as

\[
\dot{v}^*(t) = \frac{v^*(t) - x(t)}{\lVert v^*(t) - x(t) \rVert} \tag{2}
\]

However, due to the delay \( \tau \), discussed earlier, along with the imperfection of the map \( f \), the motor command needs a correction before introducing it to the robot. Thus, the cerebellum acts as a forward model to predict spatial velocities upon applying \( u \). While the robot motion can be expressed as

\[
v(t) = g(q(t), u(t - \tau)), \tag{3}
\]

where \( g \) represents the actual differential kinematics of the robot, and \( v(t) \) is the spatial velocity of the end-effector upon executing the command \( u(t - \tau) \) while the robot joint configuration is \( q(t) \).

The spatial velocity predictions generated by the cerebellum \( \hat{v}(t) \) can then be expressed as

\[
\hat{v}(t) = \hat{g}(q(t), u(t - \tau), v(t), v^*(t)), \tag{4}
\]

where \( \hat{g} \) represents the forward model built by the cerebellum to approximate the map \( g \). The model approximations are improved during training relying on the feedback error \( e \), where \( e(t) = v(t) - \hat{v}(t) \). Hence, the error in sensory signals and discrepancy from the predicted value is used to modulate the desired spatial velocity before introducing to the DM, as detailed later in Sec. 3.3. So, DM makes use of the predictions provided by the cerebellum to correct the anticipated error in sensory readings due to the dead-time effect:

\[
u(t) = f(q(t), v^*(t), \hat{v}(t), \hat{v}(t + \tau)). \tag{5}
\]

Taking into consideration both the error from previous trials and the sensory prediction, the controller is enabled to correct the next motor command in an indirect way. This is carried out by adding the error from previous attempts to the predictions of the robot state to make up for the expected error:

\[
\hat{v}(t) = \hat{v}(t + \tau) + e(t). \tag{6}
\]

Finally, the corrected prediction \( \hat{v}(t) \) is compared to the desired velocity \( \dot{v}^* \):

\[
\hat{v}(t) = 2\dot{v}^* - \dot{v}(t). \tag{7}
\]

To put it in other words, \( \hat{v} \) is the sensory signal that can be introduced to the DM to give better estimations and make up for the delay in the feedback cycle.

\[
u(t) = f(q(t), \hat{v}(t)). \tag{8}
\]

### 3.2. The differential mapping SNN

A two-layer SNN, one input and one output layer, are connected via all-to-all plastic synapses (weights can change) to provide the transformation between two correlated spaces. The network encodes the current joint angles and the spatial velocity of the end-effector in the input layer and encodes the angular
velocity of the joints in the output layer. As the network correlates mainly the velocities in two-spaces, it acts as a differential map; it is named as Differential Mapping Spiking Neural Network (DMSNN). For a robotic manipulator of $n$ degrees of freedom (DOF), TS is represented by $n$ assemblies of neurons, where each assembly encodes a corresponding dimension.

Similarly, for an $m$ DOF of JS, $m$ assemblies are needed. Hence, the input layer is made up of $n+m$ assemblies. The assemblies $F^{n+m}$ encode the $n$-dimensional spatial velocity, while the assemblies $P^{m}$ encode the $m$-dimensional joints' angular positions. At the output, the assemblies $F^{m}$ encode the $m$-dimensional joints' angular velocities as depicted in Fig. 3. Two layers of the network are connected via all-to-all plastic synapses (both excitatory and inhibitory). The inhibitory synapses regulate the motor (output) neurons activity to maintain stable learning and hence avoid the unbounded increase in the weights of the excitatory synapses. At the output layer, local inter-inhibitory connections (i.e., within each assembly) are added with low inhibition to proximal neurons and higher inhibition to the distal neurons.

Modulation of the plastic synapses occurs during training the network, as shown in Fig. 4, where random target angles $\theta_k$ are generated for motor babbling. The robot joints move towards $\theta_k$ linearly in JS based on the error calculated as the difference between the desired joint angles and current one $q$. The internal (i.e., proprioception) and external sensors (i.e., exteroception) provide the necessary data to both the sensory and motor layers while training DM. These variables are encoded based on the preferred/central value $\psi_i$ defined for each neuron, and a distribution that allows all the neurons in the assembly to contribute to what is known as population coding. The neurons’ firing rates are defined by the Gaussian tuning curve which can be expressed as

$$\Theta_i(l) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{l - \psi_i^2}{2\sigma^2}\right),$$

where $\psi$ is the variable’s value, and $\sigma$ is the radius calculated based on the number of neurons per assembly $N_i$ and the defined range of values of each variable. The synaptic weights are modulated accordingly forming a proper DM. After training ends, the control phase starts where DM is ready to execute a coarse robotic servoing task. The corresponding values of the variable are encoded and introduced to assemblies $P^{m}$ and $F^{m}$, and the output is then decoded from the activity of $F^{m}$. The decoding scheme in this case is the central neuron

$$\psi_{out} = \frac{\sum \Theta_i}{\sum \Theta_i}$$

where $\psi_i$ is the defined central value of neuron $i$ in $F^m$ assembly, and $\psi_{out}$ is the decoded/estimated output value. However, DM is formed as a coarse map which lacks in both the precision and accuracy needed for fine control of the robot.

3.3. The proposed forward cerebellar-like model

The cerebellum is composed of three layers. The Granule layer, which is the innermost layer, is made up mainly of granule cells, which count up to 80% of the neurons in the brain. It contains as well the Mossy fiber axons and the Golgi cells. The Purkinje

Fig. 3. Input (sensory) neurons are connected to output (motor) neurons through excitatory and inhibitory plastic synapses. The signals introduced to each neuron assembly are depicted. The motion is guided during the training of the network through motor babbling actions in JS based on motor commands $u_k$. During training, the values of angular positions $q$, angular velocities $\dot{q}$, and spatial velocities $\dot{v}$ are encoded and fed to the corresponding assemblies. During the control phase, encoder readings $q$ and updated desired spatial velocity $\dot{v}$ are encoded and fed to the input layer, and the robot is controlled by decoding motor commands from the activity in the output layer. For arrows between the input and output layers, the hollow arrow heads refer to plastic synapses, while solid ones refer to nonplastic synapses.
layer is the middle layer and contains the Purkinje cells which are featured by the distinctive firing pattern and considered a key component for learning to occur in the cerebellum. The **Molecular layer** is the outermost layer containing the axons extended from the granule cells to purkinje cells (known as parallel fibers) intersecting with axons extended from the inferior olive (known as climbing fibers). The molecular layer also contains the basket and stellate cells.

The cerebellar computational model developed, as shown in Fig. 4, acts to rectify the sensory readings before introducing to the DM which replicates the function of the motor cortex. This enhances the motor commands generated by DM by accounting for sensory discrepancies in the previous cycles as explained in the two previous subsections. The cerebellar forward model is developed based on the cerebellar microcircuit.

The **Mossy Fibers** (MF) encode the current angular position of the joints and the desired Cartesian velocity. These two variables are chosen to provide the essential information to define the state of the robot in both JS and TS. Thus, MF consists of $n_{JS}$ assemblies of neurons, and each assembly encodes a dimension of one of the variables. The angular position is encoded by $n_{JS}$ assemblies forming the MF$_{JS}$ group, where $n_{JS}$ is the number of DOFs studied. The desired Cartesian velocity is encoded by $n_{TS}$ assemblies forming the MF$_{TS}$ group, where $n_{TS}$ is the number of Cartesian DOF studied. The synapses connecting between MF, Granular cells (GC) and Golgi cells (GgC) shall lead to a sparse coding of robot states. The Inferior olive (IO) provides the teaching signal, which in this study is the TS error, through the climbing fibers (CF) to Purkinje Cells (PC). The error ($e$) is defined as the discrepancy between the actual and the predicted spatial velocities. The plastic synapses connecting GC to PC, known as parallel fibers PF, are modulated initially under the effect of the teaching signals from CF. These signals evoke activity in PC to provide the desired correction, and thus allows to encode such corrections at the corresponding state of the robot.

The MF connects to Deep Cerebellar Nuclei (DCN) through excitatory synapses to maintain a basal spiking activity. The PC connects to DCN through inhibitory synapses to allow for the right value to be decoded from the activity of these neurons, while IO connects to DCN through excitatory plastic connections studied to provide fast convergence of learning.

The PC, and similarly IO and DCN, consists of $n_{TS}$ neurons’ groups with each group consisting of two assemblies for positive and negative change for each DOF.

After the DM training goes for several iteration, i.e. till the coarse control map is formed, the training then starts for the cerebellum to build the corrective mapping.

In Ref. 39, a study was conducted on infants between 6–12 months to monitor the neural activity in the motor cortex (M1) while reaching targets. It was observed that the activity shifts from a diffused state across M1 to a focused one as the age of infants increases. Additionally, insufficient data is reported about the development of the cerebellum in infants at that age. Thus, it is safe to assume that in infants the development of the cerebellum starts later than the motor cortex and its contribution increases with...
the increase in its size which is reflected by performing fine movements and exhibiting some motor skills.2

**Encoding in MF.** Similar to encoding in DM, the input values to MF is first encoded employing population coding, with the current introduced to the input values to MF is first encoded employing population coding, with the current introduced to the

\[ \Theta(i,j) = \exp \left( -\frac{\| i - j \|^2}{2\sigma_{MF}^2} \right) \]  

where \( \Theta(i,j) \) is the input to the \( j \)th MF assembly, and \( \sigma_{MF} \) is the radius for the Gaussian distribution, and the variable ranges from \( \theta_{MF_{max}} \) to \( \theta_{MF_{min}} \), \( \theta_{MF_{i,j}} \) is the preferred/central value of \( i \)th neuron in \( j \)th MF assembly. Both \( \theta_{MF_{i,j}} \) and \( \sigma_{MF} \) are adjusted using the self-organization algorithm (SOA), where data previously collected for babbling \( \Xi \) is used to adjust the values of \( \theta_{MF_{i,j}} \) for all neurons. SOA: The algorithm is an adaptation of self organizing maps to reorganize the linearly initialized set \( \theta_{MF_{i,j}} \) (i.e. equally spaced from \( \theta_{MF_{min}} \) to \( \theta_{MF_{max}} \)) for better encoding based on the density of data in specific regions. It was chosen for the biologically appealing properties which fits with the theme of this study. The algorithm depends on three concepts: competition, cooperation, and adaptation.3,4

A best matching unit \( \beta \) is chosen for each central value from the set \( \theta_{MF_{i,j}} \) based on the competition with other units/neurons. \( \beta \) is picked based on the Euclidean distance from a random data sample \( \xi \) (from the set \( \Xi \)):

\[ \beta = \arg \min_{i,j} \| \xi - \theta_{MF_{i,j}} \|^2 \]  

where the value of \( \theta_{MF_{i,j}} \) and that of the neighboring units, emphasizing the cooperation concept, at an instant \( k \) are updated (i.e. adaptation) such that

\[ \theta_{MF_{i,j}}(k+1) = \theta_{MF_{i,j}}(k) + \rho(k) \nu_{ij}(k) \| \xi - \theta_{MF_{i,j}}(k) \| \]  

where \( \rho \) is the learning rate and \( \nu \) is the neighboring/proximity function given by

\[ \nu_{ij}(k) = \exp \left( \frac{-\| i - j \|^2}{2\sigma_{ij}^2} \right) \]  

and \( \theta \) is the radius gauging the proximity of the neighborhood. Both \( \rho \) and \( \sigma \) decay exponentially over the whole period \( K \):

\[ \rho(k) = \rho_0 \exp \left( -\frac{k}{K} \right) \quad \sigma(k) = \sigma_0 \exp \left( -\frac{k}{K} \right) \]  

where \( \rho_0 \) and \( \sigma_0 \) are the initial values for \( \rho \) and \( \sigma \), respectively. After running the SOA for \( K \) iterations, the radius \( \sigma_{MF_{i,j}} \) is set for every neuron in MF separately such that

\[ \sigma_{MF_{i,j}} = \sigma_{MF_{i,j}}(n - 1) \]  

with the last radius (i.e. \( \sigma_{MF_{i,j}}(N) \)) set to have the same radius as the previous one (i.e. \( \sigma_{MF_{i,j}}(N-1) \)).

**Sparse encoding in GC.** The application of the SOA to MF allows for a more distinctive input to the GC and better encoding of the robot states. Both GC and MF are connected through excitatory synapses to the Golgi Cells (GgC), which is connected to the GC through plastic inhibitory connections. The recurrence through the reciprocal connections between GC and GgC along with the connections to the PC allows for sparse encoding of the robot state space and can interpreted as a Liquid State Machine (LSM) as argued in Ref.22. It is also claimed that the inhibitory action of GgC allows to have a minimum number of neurons in GC active at the same time and thus higher sparsity and better encoding of the states.22

Each neuron from GC is connected to a randomly picked neuron from each assembly MF, and hence each neuron from GC shall be connected to \( n_{MF} \) neurons. Furthermore, MF connects via excitatory synapses (all-to-all connections) to DCN. Synaptic connections to PC. GC connects to PC via plastic excitatory projections (which are known as Parallel Fibers (PF)). The parameters of the network shall be tuned such that PF are modulated only when IO neurons are active. The projections from IO to PC, known as Climbing Fibers CF, are one-to-one synapses to ensure that neurons belonging to the same group (i.e. the same DOF) and direction (i.e. positive/negative changes) connect to each other, and hence ensuring that the right members of CF are modulated. IO connects through excitatory synaptic connections to DCN. The activity of
neurons in IO is given by
\[
\theta_{\text{IO}+} = \begin{cases} 
\theta_{\text{IO}+max} & \text{if } \epsilon_{\text{pred}} > \Upsilon, \\
0 & \text{if } \epsilon_{\text{pred}} \leq \Upsilon,
\end{cases}
\]
(17)
\[
\theta_{\text{IO}−} = \begin{cases} 
0 & \text{if } \epsilon_{\text{pred}} \geq -\Upsilon, \\
\theta_{\text{IO}−max} & \text{if } \epsilon_{\text{pred}} < -\Upsilon,
\end{cases}
\]
(18)
where \(\theta_{\text{IO}+}\) and \(\theta_{\text{IO}−}\) describe the mean firing rates of the two opposite directions of the \(j\)th DOF encoded by the IO assemblies, while \(\theta_{\text{IO}+max}\) describes the maximum firing rate of neurons in IO. A threshold value \(\Upsilon\) is defined to avoid an overlapping oscillatory activity around the reference value. PC \(\rightarrow\) DCN: Similarly, the connections between PC and DCN follow the same concept to allow for activation of the right group of neurons. The cerebellar output is decoded from the activity of DCN neurons:
\[
\hat{v}_j = \frac{\sum_{\text{DCN}, j} \theta_{\text{DCN}, j} - \theta_{\text{DCN}−max} \times \epsilon_{\text{max}, j}}{\theta_{\text{DCN}−max} \times \epsilon_{\text{max}, j}},
\]
(19)
where \(\hat{v}_j\) gives the anticipated/predicted velocity for the \(j\)th DOF, \(\theta_{\text{DCN}, j}\) and \(\theta_{\text{DCN}−max}\) are the firing rates of the \(i\)th neuron in the two opposing directions assemblies of the \(j\)th DOF, \(\epsilon_{\text{max}, j}\) is the number of neurons in each DCN assembly. \(\theta_{\text{DCN}−max}\) is the maximal firing rate observed in DCN, \(\epsilon_{\text{max}, j}\) defines the maximum speed for the \(j\)th DOF.

### 3.4. Optimization of the network parameters

To have the cerebellar microcircuits employ more features of the biological counterparts, an optimization process is applied to finely tune the parameters. The optimization allows to set the values of the hyperparameters \(H\) of the network to meet specific goals for that mean. Such goals are set to tune the layer by layer of the cerebellar model through the defined objective function \(f_j(\mathcal{H}_i)\) to be minimized, where \(\mathcal{H}_i \subset H\). Each objective function consists of a set of objectives \(O_i\) weighted by a vector \(w_i\), such that \(f_j(\mathcal{H}_i) = \sum w_i f_{ij}(\mathcal{H}_i)\). Tuning the model layer by layer allows to properly define and monitor the expected output from each layer. Moreover, this allows to simplify the optimization and avoid the complexities accompanying choosing a large number of hyperparameters to be optimized simultaneously.

**Objective 1 (Uniform firing in MF)**: The input signals coming from the DM is first introduced to the MF. Thus, the first step is to make sure that the firing pattern in MF is suitable to be introduced to the next layer. The parameters affecting the firing are the neuron parameters and the amplitude of the input current to the neurons. The criteria selected are the maximum firing rate of the neurons and the pattern of firing. For the Gaussian distribution of the input current, it is expected to have a corresponding Gaussian distribution, as well, for the firing rates in the MF layer. Thus, a Gaussian distribution fitting, via maximum likelihood estimation, is applied to the number of spikes released from the neurons to compare the mean value of the curve \(G_{\text{mean}}\) to the expected output (i.e. the neuron MF output whose central value is the closest to the input value). Hence, the two objectives are defined \(O_1 = [C_1; C_2]\) and can be formulated as
\[
C_1 = \sum_{n=1}^{N_{\text{test}}} \left| f_{\text{MF}}(n) - f_{\text{MF,ideal}}(n) \right| / N_{\text{test}}
\]
\[
C_2 = \sum_{n=1}^{N_{\text{test}}} \left| G_{\text{mean}}(n) - G_{\text{mean,ideal}}(n) \right| / N_{\text{test}}
\]

With \(w_1 = [0.5, 0.5]\), the first objective function is defined as \(f_1(\mathcal{H}_i) = C_1 w_1\).

**Objective 2 (Sparsity in GC)**: The output from MF is then introduced to the GC. As mentioned in the previous subsection, GC and GGC act together to give a distinctive output for each input from the MF, and thus allowing for sparse coding of the input. Consequently, \(f_2\) is designed to ensure that a minimum number of GC neurons is active and checks as well for the uniqueness of the activity pattern for every input. Additionally, the firing rate is defined to resemble the neuron activity in the biological counterpart. With these four objectives to be satisfied, \(O_2 = [C_1; C_2; C_3; C_4]\), the firing rates are defined in a way similar to that in MF, where
\[
C_3 = \sum_{n=1}^{N_{\text{test}}} \left| f_{\text{GC}}(n) - f_{\text{GC,ideal}}(n) \right| / N_{\text{test}}
\]
\[
C_4 = \sum_{n=1}^{N_{\text{test}}} \left| G_{\text{mean}}(n) - G_{\text{mean,ideal}}(n) \right| / N_{\text{test}}
\]

The objective \(C_3\) targets minimizing the number of active neurons as much as possible, but also ensures that there is still active neurons in GC (i.e. at least one neuron is active). During each iteration, the number of spikes triggered by each neuron is recorded in the set \(X_{\text{GC}}\) after the nth test trial, to be used to...
define the firing rates and the active neurons as well. Neurons with a firing rate greater than third of the desired firing rate (i.e., firing rate greater than \(0.33R_{\text{desired}}\)) are kept and the rest are discarded, then \(S_n^{\text{GC}}\) is updated accordingly. This allows to consider only the neurons that would affect the learning in PC. The difference between the optimal number of firing neurons (chosen as 1 in this case) and the number of active neurons \(S_n^{\text{GC}}\) is recorded, and then, the number of repetitions of each neuron in the set \(n\) is computed to formulate \(A_n^{\text{GC}}\) as:

\[
A_n^{\text{GC}} = \begin{cases} 0, & |S_n^{\text{GC}} - 1| \\ \frac{|S_n^{\text{GC}} - 1|}{N_{\text{test}}} \end{cases}, \quad (22)
\]

The objective \(O_3^1\) describes the uniqueness of the output obtained across the \(N_{\text{test}}\) trials, where the repetition of spiking of a neuron across many trials is penalized. To simplify the computations, from each trial \(n\) the neuron with maximum firing rate is recorded, and then, the number of repetitions of each of the neurons in the set is computed. The mean \(\mu_{\text{mean}}\) and maximum \(\mu_{\text{max}}\) number of repetitions is finally computed to formulate \(O_3^2\) as:

\[
O_3^2 = 0.4\mu_{\text{max}} + 0.7\mu_{\text{mean}}. \quad (24)
\]

With \(w_3^2 = [0.1, 0.1, 0.2, 0.6]\), the second objective function is defined as \(f_3(\mathcal{H}_3) = O_3^2\).

**Objective 3 (Proper firing rates in PC).** The neurons of PC are known to have distinguishable firing patterns in response to different inputs. The input from GC tends to trigger simple spikes (SS), while the input from IO/CF triggers complex spikes (CS) in PC. These two spiking patterns differ from each other in many aspects, but in this study only their respective firing rates are considered. Moreover, the assemblies of neurons within each group tend to have an alternating activity for different directions of motion. Thus, three objectives are defined, \(O_3 = [O_3^1, O_3^2, O_3^3]\). The first two objectives can be formulated as:

\[
O_3^1 = \frac{\sum_{n=1}^{N_{\text{test}}} |f_{\text{PC}}(n) - \bar{f}_{\text{PC}}|}{N_{\text{test}}}, \quad (25)
\]

\[
O_3^2 = \frac{\sum_{n=1}^{N_{\text{test}}} |f_{\text{PC}}(n) - \bar{f}_{\text{SS}}|}{N_{\text{test}}}, \quad (26)
\]

To construct the formula to describe the third objective \(O_3^3\) firstly the activity of the PC is compared to a threshold value (chosen as the maximum firing frequency of the simple spikes) with those above and below the threshold assigned as active/true and inactive/false, respectively. In this case, the most desirable state is to only have one group of neurons active per DOF, which is represented by an XNOR logic gate:

\[
A(\text{PC}, j) = \begin{cases} 1, & f_{\text{PC}} > f_{\text{SS}} \\ 0, & f_{\text{PC}} \leq f_{\text{SS}} \end{cases}, \quad (27)
\]

where \(f_{\text{PC}}\) is the mean firing rate of neuron in either of the two assemblies of neurons for each DOF \(j\). \(O_3^3\) can then be expressed as:

\[
O_3^3 = \frac{1}{N_{\text{test}}} \sum_{j=1}^{N_{\text{output}}} A(\text{PC}, j) \odot A(\text{PC}, j). \quad (28)
\]

With \(w_3^3 = [0.2, 0.2, 0.0, 0.6]\), the third objective function is defined as \(f_3(\mathcal{H}_3) = O_3^3\).

**Objective 4 (Proper output from DC).** After optimization is carried for the previous hyperparameters, this last objective function is directed to test the operation of the network while optimizing the parameters affecting the activity in the DCN. The robot, in a simulation environment, repeats a chosen motion toward a target four times, and both the error while moving \(\epsilon_{\text{error}}\) and the execution time \(\Delta\) are recorded. The error in this case is formulated as:

\[
\epsilon_{\text{error}} = \arccos \left( \frac{x \cdot y}{\|x\| \|y\|} \right). \quad (29)
\]

Consequently, \(O_4\) consists of six weighted objectives, where \(O_4 = [O_4^1, O_4^2, O_4^3, O_4^4, O_4^5, O_4^6]\). The objective \(O_4^1\) acts to keep the mean value of the error \(\epsilon_{\text{error}}\) across the four trials as minimum as possible, and an inverted firing pattern in the opposing groups of neurons within the DCN and when compared to PC as well. The objectives \(O_4^2\) and \(O_4^3\) are set to promote the decrease in the error \(\epsilon_{\text{error}}\) and the execution time \(\Delta\), respectively, as the training proceeds. To achieve this, each of these objectives is assigned a value of 1 at the beginning of the training, and the variables (mean value of \(\epsilon_{\text{error}}\) and \(\Delta\)) are compared to those from the previous trial, to deduct 0.33 in case of a decrease in the value of the variable. Hence, in case of a consistent decrease in the error and execution time from one trial to another, reflecting a successful
learning process, the values of these objectives would return a value zero, which is the absolute minimum in this case. The objective $O_1^f$ adjusts the firing rate of DCN neurons in the desired range such that

$$O_1^f = \sum_{n=1}^{N_{DCN}} |f_{DCN}(n) - f_{DCN,n}^{\text{desired}}|.$$  

(30)

Similar to PC, in DCN the opposing groups of neurons shall be set to fire in an alternating manner, which is formulated as

$$A_{DCN,i} = \begin{cases} 1, & f_{DCN,i} > f_{DCN,i}^{\text{desired}} \\ 0, & f_{DCN,i} \leq f_{DCN,i}^{\text{desired}} \end{cases}.$$  

(31)

where $f_{DCN}^i$ is the mean firing rate of DCN, and $f_{DCN,i}^{\text{desired}}$ is the mean firing rate of neuron in either of the two assemblies of neurons for each DOF $i$. $O_1^f$ can then be expressed as

$$O_1^f = \sum_{i=1}^{N_{DOF}} A_{DCN,i} \odot A_{DCN,i}.$$  

(32)

Additionally, the objective $O_2^f$ ensures that the activity in the assemblies of DCN opposes that of the corresponding ones for same DOF in PC (i.e. fire in an inverted manner):

$$O_2^f = \sum_{i=1}^{N_{DOF}} A_{DCN,i} \odot A_{PC,i}.$$  

(33)

With $w_4^1 = [0.3, 0.1, 0.1, 0.1, 0.2, 0.2]$, the fourth objective function is defined as $f_{DCN}(4) = f_{PC}(4)$. It can be concluded that: (i) $O_1^f$ aims to optimize the firing properties of neural groups and ensure that each group executes the assigned role. (ii) $O_2^f$ involves directly minimizing the error in predictions $\epsilon_{\text{pred}}$ (as defined by subobjectives $O_2^f$) along with optimizing the firing properties of DCN. For a proper neural robotic embodiment, both the biological features and the robot-environment interaction are essential to create a basis for neuroscientific studies.

3.5. Bayesian optimization for the objectives

To meet these objectives, Bayesian Optimization (BO) is utilized to optimize the defined objective functions BO. These functions would be very costly and time consuming to optimize through manual or random searching methods due to the high dimensionality and stochasticity of the search space. BO develops a probabilistic model for the objective function to facilitate the evaluation of the objective function while making use of the history of previous trials to guide the optimization process, as explained later in this subsection. Thus, the optimal solution is sought to minimize each objective function to obtain the optimal hyperparameters $h^*$, such that

$$h^* = \arg \min_{h \in \mathcal{H}} f(h).$$  

(34)

The main constituents through which a BO method is identified are the regression model and the acquisition function. The probabilistic regression model surrogates the objective function (and referred to usually as the surrogate model). This probabilistic model is initiated with some random evaluations to guide the algorithm, starting from complete uncertainty (prior), and develops as more evaluations are stored in the history to give better future evaluations and decrease the uncertainty (posterior). The surrogate model is expressed as $h_i = \mathcal{L} \left( \mathcal{H}_i \right)$ representing the mapping of the $\mathcal{H}_i$ hyperparameters to a probability of a loss/score $\mathcal{L}$ for an objective function $f_i$. The acquisition function (also known as selection function) allows for selecting appropriate candidates to improve the surrogate model while exploring for the optimum values for the hyperparameters. Hence, a proper choice of the acquisition functions guarantees a balance between exploration and exploitation without getting trapped in a local minima. In this study, an Adaptive Tree Parzen Estimator (ATPE) is chosen as a regression model and the Expected Improvement (EI) as the acquisition function.

The standard TPE is known to fit for optimization problems where either a mixture of discrete and continuous hyperparameter spaces are to be studied or when the hyperparameters are contingent upon each other, and it is chosen for the latter reason. In TPE, rather than describing the posterior $P(\mathcal{L}|\mathcal{H})$, it describes instead $P(\mathcal{H}|\mathcal{L})$, relying on Bayes rule such that

$$P(\mathcal{L}|\mathcal{H}) = \frac{P(\mathcal{H}|\mathcal{L})P(\mathcal{L})}{P(\mathcal{H})}.$$  

(35)

The TPE targets building two separate hierarchical processes, $P(\mathcal{H}|\mathcal{L} \in \mathcal{U})$ and $P(\mathcal{H}|\mathcal{L} \in \mathcal{D})$, where the sets $\mathcal{U}$ and $\mathcal{D}$ contain the highest and lowest values of $\mathcal{L}$, respectively, observed so far reference to a defined threshold value $L^*_i$. This threshold value
A Neurorobotic Embodiment for Exploring the Dynamical Interactions of a Spiking Cerebellar Model

is decided based on a predefined percentage $\gamma$ such that $P(L_i < L^*) = \gamma$. The likelihoods $U$ and $D$ are modeled via kernel density estimators (which in this case is the Parzen estimator). The Parzen estimator PE allows to represent a function through a mixture of kernels $K$, which are continuous distributions, to be expressed as

$$P(H_i) = \frac{1}{N_p} \sum_{j=1}^{N_p} K(H_i - H_j),$$  

(36)

where $N_p$ is the number of kernels used for the approximation, $\eta$ is the bandwidth of each kernel, and $K$ is chosen to be a normal distribution. Modeling $U$ and $D$ gives a way to choose hyperparameters for the next observations that are more likely to return lower values for the objective functions (in the case of minimization of objective functions).

Although TPE has less time complexity compared to other BO methods (as Gaussian Process BO), TPE does not model interaction/correlations among the hyperparameters. The ATPE addresses this drawback by judging from Spearman correlation between the hyperparameters which parameters to vary and which parameters to lock to achieve a more efficient exploration. Also, among the drawbacks of TPE is that it has a fixed value for $\gamma$ and a fixed number of candidates introduced to the acquisition function to predict the next candidate optimal solution, which were introduced initially while solving some specific problems. ATPE introduced empirically concluded formulas based on the cardinality of the search spaces for the hyperparameters to give better values for these two variables.

In this study, the Expected Improvement EI is chosen as the acquisition function, to maximize the ratio $P(N|L_i \in D)/P(N|L_i \in U)$. The EI generates a probability of obtaining a better solution than the current optimum solution and the amount of expected improvement as well, and consequently, favors bigger improvements. Thus, the EI outperforms the maximum probability of improvement, which estimates the probability of an improvement to occur, but not the amount of improvement. Also, EI was found to outperform upper/lower confidence bound as compared in Ref. [22]. The basic formula for the EI is

$$EI_{L_i}(H_i) = \int_{H_i}^{\infty} (L_i - L_j) P(L_j|H_i) dL_j,$$  

(37)

By applying Bayes rule (Eq. (35)) and substituting in Eq. (38) the EI can be written as

$$EI_{L_i}(H_i) = \frac{2L_i^2 P(H_i) - D(H_i)}{\gamma D(H_i) + (1 - \gamma) E(D(H_i))},$$  

(38)

$$EI_{L_i}(H_i) \propto \left(\gamma + \frac{H/H_i}{D(H_i)} (1 - \gamma)\right)^{-1}.$$  

(39)

From formula (38), it can be concluded that EI acts to maximize the ratio $D(H_i)/U(H_i)$ and thus leading to introducing better candidates for the next search while still maintaining a trade-off between exploration and exploitation.

3.6. Neuron model

To model the spiking activity of the neurons in the cerebellum in real-time, the simple neuron model developed by Izhikevich is chosen for its ability to reproduce different firing patterns. The model provides a decent biological plausibility at a relatively low computational cost. The following differential equations describe the model:

$$\dot{V} = f(V, U) = 0.04V^2 + 5V + 140 - U + I,$$  

(40)

$$\dot{U} = g(V; U) = a(V^2 - U).$$  

(41)

Additionally, the membrane potential is reset after triggering a spike such that

$$V \geq 30 \text{ mV},$$

then

$$V \leftarrow c, \ U \leftarrow (U + d),$$  

(42)

where $V$ (in mV) is the membrane potential and $U$ is the variable that acts to lower the neuron’s membrane potential (which is also known as the recovery variable). The parameter $a$ (in ms$^{-1}$) decides the time scale of $U$ (i.e. the decay rate), and $b$ (dimensionless) describes the sensitivity of the neuron before triggering a spike (i.e. sensitivity of $U$ to the sub-threshold $V$). The parameter $c$ (in mV) describes the reset value of $V$ after a spike is triggered, and $d$ (in mV) gives the reset value of $U$ after triggering spike. The external currents introduced to the neurons are described by $I$.
learning rule is chosen. In STDP, the change in the synaptic weight depends on the spikes relative timing in both the pre and post synaptic neurons. For the CB, the antisymmetric STDP modulates the weight of the plastic synapses, and is formulated as

$$\Delta w_{ij} = \begin{cases} -S_j \exp(-\Delta t/\tau_i) & \Delta t \leq 0, \\ S_i \exp(-\Delta t/\tau_j) & \Delta t > 0, \end{cases}$$

where the two coefficients $S_j$ and $S_i$ decide the amount of the decrease and increase of the synaptic weight, respectively. $\tau_i$ and $\tau_j$ determine the time windows through which synaptic weights decrease and increase, respectively.

The symmetric STDP is applied for the plastic synapses in DM and can be formulated as

$$\Delta w_{ij} = S \{1 - (\Delta t/\tau_j)^2\} \times \exp((\Delta t/\tau_i)^2),$$

where $S$ determines the amount of change in synaptic weights, while the ratio between the $\tau_i$ and $\tau_j$ adjusts the time window for increasing and decreasing the synaptic weights. $\Delta t$ is the difference between the timing of spikes at post-synaptic and pre-synaptic neurons, respectively, such that $\Delta t = t_{pre} - t_{post}$.

4. Results

4.1. Setup

A UR3 universal robot is used to test the developed cerebellar controller in two experiments. The shoulder and elbow joints are controlled in an experiment to test the manipulation of the end-effector to a desired position as shown in Fig. 5(a), while the elbow and wrist joints are controlled to test the manipulation of a deformable object, as shown in Fig. 5(b). The shoulder and elbow joints for the first experiment are defined to have ranges of $q_S = [-180^{\circ}, -135^{\circ}]$ and $q_E = [-210^{\circ}, -180^{\circ}]$, respectively.

While the elbow and wrist joints for the first experiment are defined to have ranges of $q_E = [-45^{\circ}, -20^{\circ}]$ and $q_W = [-210^{\circ}, -180^{\circ}]$, respectively. A camera is fixed on top of the robot to track the end-effector position and the deformable object through color filtration. The proprioceptive readings (i.e. joints’ positions and velocities) of the robot joints are obtained from motor encoders. The information required for motor babbling is recorded by giving commands to move to random joint angles linearly in JS within the defined ranges, through a script-based programming language developed for universal robots.

For the end-effector manipulation task, the robot is instructed to move to only 100 points in the JS and the collected data during motion is then used to train DM, while for the deformable object case the robot is instructed to move to 300 points as small increments in joint angles may lead to a big change in the centroid of the deformable object and hence more rich data is needed. The spiking neural networks are developed using NeMo package which allows simulating the network using GPU. In this study, a computer with i7-6700K CPU and a GeForce GTX 1080Ti is used to build the networks and control the robot. Each iteration takes 80 ms, where the GPU allows running the simulation in real-time (i.e. 1 s in simulation is equal to 1 s in real world clock). The simulation can be accelerated by using a more powerful GPU or a neuromorphic chip, however 80 ms allows a proper averaging of the output for a moderate movement speed without jerky motions.

4.2. Optimization results

After running the optimization described in the two previous subsections, the chosen parameters are obtained as shown in Table 1. The optimization is run on a robot in the simulation environment to avoid wear of the mechanical parts and for safety, but the final experiments are conducted on a real robot. Neurons in MF achieve a maximum firing rate of 62 Hz which is comparable to 60 ± 35 Hz spontaneous firing rate observed at excitatory synapses connecting MF to GC, and an average firing rate (for active neurons only) of 40 ± 6 Hz compared to 20 ± 2 Hz. The activity in each assembly is shown in Fig. 3(a). For the neurons in GC the sparsity is achieved by having 6 ± 3 neurons only active at the same time for a certain input, while maintaining a maximum firing...
where analysis of the PC properties in Ref. 61 proved bistable neurons demonstrated by Izhikevich, activity and thus limiting number of active neurons. between GC and GgC, to suppress the undesirable plasticity in synapses locomotion state. The sparse activity is achieved by in Ref. 59 for maximal firing rate in GC while at ± rate of 86 Hz compared to 106 ± 65 Hz reported in Ref. [33] for maximal firing rate in GC while at locomotion state. The sparse activity is achieved by the aid of firing in GgC and the plasticity in synapses between GC and GgC, to suppress the undesirable activity and thus limiting number of active neurons.

The parameters of PC neurons are close to that of bistable neurons demonstrated by Izhikevich [33], where analysis of the PC properties in Ref. [33] provides indications of bistable behavior, as shown in Fig. (b). Hence, the neurons in PC fire simple spikes with a firing rate of 70 ± 7 Hz (i.e. when no input is introduced from IO through the CF) which is comparable to an average of firing rate of 50 Hz. In case of complex spikes a firing rate of 160 ± 14 Hz is achieved, which is mentioned to achieve in biological counterparts to firing rates up to 400 Hz. While the modeled PC fire slower than the real cells, however the big difference in the firing rates of simple and complex spikes facilitates the choice of adequate learning parameters for proper modulation of the PF synapses. It shall be noted from Fig. (b) that higher firing rates are achieved in the beginning, corresponding to bigger errors (and thus higher activity in IO), then the rate of activity decreases. After several trials, the strength of PF synapses increases, and hence the firing rate increases. The biostability achieved in PC relies on the inhibitory activity from BC to be able to adjust the firing rates. While BC is not included in the optimization objectives, the obtained parameters refer to the fast spiking activity which suits its interinhibitory role in the model.
Fig. 7. (a) The error in cerebellar predictions $e_{pred:a}$ as defined in Eq. (29), as the learning proceeds for four trials, (b) the average firing rates of IO and DCN, (c) the phase portrait and (d) firing rates of PC.

spikes generated by IO is characterized by a low rate ($<3\text{Hz}$) as seen in Fig. 7(b), which is comparable to a firing rate 1 Hz reported,64 however a large number of CF synapses connects between each neuron of PC to a corresponding neuron of IO. This allows this low frequency stimulus to trigger high frequency spikes in PF. The maximum firing rate in DCN is $40 \pm 8\text{Hz}$, which is comparable to the instantaneous firing rate $30 \sim 50\text{Hz}$ reported in Ref. 65.

4.3. Radial reaching

In Ref. 66, patients with cerebellar damage attempt to reach radially towards targets at the same distance starting from the same central point with the aim of testing the smoothness of movements of these patients and joint motion coordination compared to healthy persons. Similarly, the robot in this study is commanded to reach radially towards target points equally distributed in an eight-angled star shape with a $45^\circ$ internal angle and distant from the center 7 cm each. It can be concluded from the robot movements in the eight directions while relying only on DM in the left panel in Fig. 8(b) that always the biggest error is presented in the first generated motor commands while the robot moves from rest. Thus, a small distance is defined for reaching to demonstrate the subsequent effects of the from-rest estimation error. The cerebellar network training is done by repeating the reaching motion to each of the eight targets only six times with an improvement in the reaching motion as shown in the right panel in Fig. 8(b) compared to what obtained after only three repetitions as shown in the middle panel. The maximum deviation recorded for repeating the reaching movement (after training ends) 10 times in each direction is reduced from ca. 21 mm to ca. 8 mm, and reduced by a mean value of ca. $310\%$ (from ca. 11.8 mm to 3.8 mm), while the time for executing reaching movements is reduced by a mean value of ca. $235\%$ (from ca. 26 s to ca. 11 s). The training and testing are done one by one for each direction. This outperforms the network developed in Ref. 17 which can be considered as a suboptimal cerebellar network compared to the
A Neurorobotic Embodiment for Exploring the Dynamical Interactions of a Spiking Cerebellar Model

network developed in this study where less biological similarity is achieved through trial and error tuning of the hyperparameters. The suboptimal network developed in Ref. 17, as shown in Fig. 9(b), improved the output from DM by only a reduction of 35% in the maximum deviation and 120% in execution time, after training for eight repetitions. Thanks to the added neuron groups, GgC and BC, better similarity to firing properties and more precise predictions are achieved with the aid of the hyperparameter optimization technique utilized.

4.4. Deformable object manipulation

To demonstrate the ability of the developed cerebellar controller to facilitate learning different skills, an experiment is set for the robot to manipulate a deformable object. The contour of the deformable object is observed based on the color, and the moments are calculated such that

\[ M_{ij} = \sum_{x_{1}, x_{2}} x_{1}^i x_{2}^j \varphi(x_{1}, x_{2}) \]

where \( \varphi \) gives the intensity of pixels. The centroid \( C = (C_{x}, C_{y}) \) is then calculated in pixels based on the moment such that \( C_{x} = M_{00}/M_{01} \) and \( C_{y} = M_{10}/M_{01} \) as explained in Ref. 17.

The centroid of the object is then converted to the world coordinates using the intrinsic parameters of the camera:

\[ x_{1} = (C_{x} - P_{x}) \frac{F}{x_{3}} \]
\[ x_{2} = (C_{y} - P_{y}) \frac{F}{x_{3}} \]

where \( C_{x} \) and \( C_{y} \) denote to the components of the centroid position in pixels, \( x_{3} \) is the object’s depth away from the camera, \( P_{x} \) and \( P_{y} \) denote the principal point and \( F \) is the focal length.

Similar to the motor babbling in case of end-effector tracking, in this case motor babbling is carried out and the corresponding value of centroid is recorded for the joint values to be used for training DM. It shall be noted that using the same parameters as those used in the previous experiment fails to develop a proper map to guide the robot. Thus, DM alone fails to drive the robot to manipulate the object properly. However, the cerebellar model and the control architecture allow to drive the robot and develop the plastic synapses properly to guide the robot motion to deform the object easily to guide the robot motion to deform the object properly as shown in Fig. 9(b). The robot is given 10 random targets in the studied work-space which are at least 5cm apart from each other, only three of which are reached using the DM with an average final error of less than 4 mm is achieved. While this study considers the centroid as one feature to manipulate the deformable object, it may not be the optimal choice to achieve such task. Other features can be explored for future studies to give better candidates to reduce the high-dimensional data needed to characterize a deformable object.

5. Discussion and Conclusions

In this study, a biomimetic control system is developed based on the detailed microcircuit of the cerebellum. A spiking cellular-level forward cerebellar model is integrated with a differential map to help improve the motion in terms of speed and deviation from the desired path. The learning is supervised by
a teaching signal based on task-based sensory feedback, in contrast to most studies in the literature relying on joint-based errors as mentioned earlier. The forward model, acting as a Smith Predictor, then compares the expected output with the actual one to build an anticipation of error for the next cycle and provide corrections to the sensory feedback to the motor-cortex-like differential mapping network, where the spatial motion plan is converted into motor commands. Both the angular and spatial accelerations are not included in this study as the robot moves at moderate speed with a light structure, hence the motion dynamics is not an effective factor.

While the optimization is carried out utilizing the simulated robot, both the data collection for DM and the testing/control phase are executed using the real robot. The optimization takes an average of 300 iterations to obtain the best candidates for each objective, which can still be handled by a real robot given that safety procedures are taken into account. The ability of the network to learn by babbling without any prior knowledge and improve the accuracy by learning from errors in TS makes the network a good candidate to control robots with complex kinematics. However, it shall be noted that a proper choice of the features learnt by the network would be crucial to reduce the amount of the babbling data collected and the time needed to learn the task properly.

The network parameters are optimized using an ATPE-based Bayesian Optimization. The optimization is carried out step by step to avoid the complexity of handling all the parameters at the same time. This allows to achieve the desirable performance while still maintaining the biological properties of the network components. This allows for future studies to use the detailed computational model to study cases with cerebellar damage in which monitoring the activity of all the neurons simultaneously is not feasible.

The obtained results show that cerebellum acts to reduce the deviation from the target path and the execution time. Additionally, it demonstrates the ability to learn new skills such as deformable object manipulation based on the error in task performance. The developed model demonstrates the ability to reduce the error in a certain direction in only few repetitions indicating the fast convergence of learning, and the suitability to be further developed for real-time adaptive robot control in multiple scenarios and applications.

Moreover, it shall be noted from the conducted experiments that the estimation error in DM is not uniform across the map. This is related to the way of collecting and introducing data for training. This is analogous to having areas mapping different body parts in the motor cortex having different density of neurons depending on how frequent and accurate are the motions generated by each body part. Hence, providing sensory corrections from CB allows DM to improve the quality of the motor output and points to the possibility of transfer of learning by having the cerebellum correcting the motion, and thus improving the quality of the training data introduced to the motor cortex. The radial reaching experiment obtained with the robot arm displays a fair similarity with the observations in Ref. 29.

This study considers only the planar motion at the end-effector as those carried out to test the motion of patients suffering from cerebellar damage. Future studies shall include nonplanar motions and incorporate dynamics/forces acting at the end-effector. Additionally, to scale up the proposed network, several considerations have to be taken into account: (i) DM scaling, similar to scaling up of MP, PC, IO, and DCN, would be simply add additional assemblies corresponding to the increased DOFs. (ii) scaling GC, and consequently GgC and BC, would depend on the size and dimensions of the defined workspace, as well as the precision needed to encode the states of the robot. (iii) for more complex mappings and larger workspaces it may be essential to divide the latter into subspaces that can be approximated by multiple forward models.

Future models would include more features, where developing highly detailed models would allow identifying cerebellar dysfunctions and studying lesions at the cellular-level, which may not be possible using current state-of-art techniques. A more detailed model for the motor cortex, with optimized hyperparameters, would be included rather than the currently simplistic model of DMSSN.

Acknowledgments

This research work was supported in part by the Research Grants Council Under Grant 14200917, in part by PROCORE-France/Hong Kong Joint
Research Scheme Under Grant F-PolyUS03/18, in part by the Key-Area Research and Development Program of Guangdong Province 2020 Under Project 76, in part by the Jiangsu Industrial Technology Research Institute Collaborative Research Program Scheme Under Grant ZG9V, and in part by PolyU Under Grants 252047/18E, ZEIJH, and UAKU.

References
36. S. Amari et al., The Handbook of Brain Theory and Neural Networks (MIT Press, 2003).
A Neurorobotic Embodiment for Exploring the Dynamical Interactions of a Spiking Cerebellar Model