Abstract—The work presented here is a novel biological approach for the compliant control of a robotic arm in real time (RT). We integrate a spiking cerebellar network at the core of a feedback control loop performing torque-driven control. The spiking cerebellar controller provides torque commands allowing for accurate and coordinated arm movements. To compute these output motor commands, the spiking cerebellar controller receives the robot’s sensorial signals, the robot’s goal behavior, and an instructive signal. These input signals are translated into a set of evolving spiking patterns representing univocally a specific system state at every point of time. Spike-timing-dependent plasticity (STDP) is then supported, allowing for building adaptive control. The spiking cerebellar controller continuously adapts the torque commands provided to the robot from experience as STDP is deployed. Adaptive torque commands, in turn, help the spiking cerebellar controller to cope with built-in elastic elements within the robot’s actuators mimicking human muscles (inherently elastic). We propose a natural integration of a bio-inspired control scheme, based on the cerebellum, with a compliant robot. We prove that our compliant approach outperforms the accuracy of the default factory-installed position control in a set of tasks used for addressing cerebellar motor behavior: controlling six degrees of freedom (DoF) in smooth movements, fast ballistic movements, and unstructured scenario compliant movements.

Index Terms—Adaptive spiking control, cerebellar modeling, compliant robotics, real-time (RT) control, spike-timing-dependent plasticity (STDP).

I. INTRODUCTION

Throughout history, human beings’ ability to interact with others has facilitated their collaboration and coordination toward achieving common goals. For many daily tasks, collaboration between humans involves physical interactions in a shared context. The emergence of humanoid robots by the mid-1990s brought a new “individual” to interact within this shared context, thus extending the human-to-human interaction theory to human–robot interaction (HRI) theory.

Efforts to physical HRI have been incrementally devoted over the last years [1], addressing new application domains in which new generations of compliant robots begin to coexist and physically interact with humans (e.g., rehabilitation therapy [2], social interaction [3], and education [4]). This new approach contrasts with the traditional well-structured industrial robotic scenarios lacking HRI. Physical HRI implies robots operating in complex unstructured environments in which human actions cannot be modeled; thus, demanding robot behavior to be autonomous, reactive under unpredicted actions, adaptive, and safe (i.e., human-like behavior) [5]. The achievement of such compliant behavior can be addressed, considering different design aspects of robotic hardware (rigid versus flexible materials, elastic actuators, low-power actuators, etc.) and software (position versus torque control, adaptive control systems, etc.).

Regarding hardware design, robots can be equipped with passive intrinsic compliance by means of different elastic components, muscle-like actuators, and/or soft materials. This approach, taking inspiration from biology, offers a compliant alternative to classical rigid-bodied robots. Yet, traditional position control methods are not of direct application in the presence of elastic materials whose mathematical modeling is almost intractable, thus demanding new control strategies [6], [7]. These traditional methods offer excellent accuracy for industrial rigid-bodied robots in well-structured environments (e.g., automated car factories), where HRI is explicitly avoided since neither safety nor compliance can be guaranteed. Compliance demands torque control, and torque control strategies based on dynamics modeling cannot be efficiently applied since the nonlinearities of elastic components make detailed modeling extremely complex [8]. Controlling biologically inspired robots carrying elastic components and low-power actuators shall directly benefit from a better understanding of biological motor control itself.

The control mechanisms encountered in biology are involved in a continuous learning process to cope with the
complexity and changes in the body structure and dynamics. Artificial intelligence (AI) can be used to replicate this learning process. In particular, widely used artificial neural networks (ANNs) have been proposed and tested as a solution for the control of these compliant robots without requiring prior knowledge of the robot dynamics [8], [9]. ANNs are vaguely inspired in the functioning of their biological neural-network counterparts. They consist of interconnected computational units, called artificial neurons, whose entry information travels from one computational unit to another across the ANN. The entry information is processed, at a neuron level, via some nonlinear function of the sum of neuron inputs and then it is transmitted through the neuron connections, that is, typically represented by a real number. Neuron connections are adjusted as learning proceeds. ANNs are designed to address the problems by considering well-structured data typically using standard analog representations for neural activity. Their lack of ability for carrying neural information via well-timed neuron action potentials (spikes) prevents them from serving as the linkage between biological neural coding and movement coordination, thus sideling any attempt at drawing biological analogies.

Spiking neural networks (SNNs) constitutes a more biologically plausible approach of neural networks. They model the information transfer and processing as occurs in biological neurons, that is, via the precise timing of spikes (discrete events at points in time) [10]. Torque control deals with the robot’s inner dynamics, that is, the evolution through time of a physical system. The SNNs temporal coding enables capturing the temporal evolution of analog sensorimotor signals [11], a pivotal feature in motor control and movement coordination [12]. SNNs intrinsic temporal characteristics make them a suitable solution for dealing with robot inner dynamics in torque control.

Several areas of the central nervous system (CNS), such as the cerebral cortex (including premotor cortex, parietal cortex, and primary motor cortex), brainstem, inferior olivary nucleus, cerebellum, and spinal cord, contribute to the temporal coordination implied in the motor control of the skeleton muscle system [13] (Fig. 1). The cerebellum stands out amongst them by its role in the integration, regulation, coordination of motor processes, and more important, motor learning [14]–[17].

Diverse computational models of the cerebellum have been proposed in robotic control (e.g., CMAC [18], APG [19], MPFIM [20], Schweighofer–Arbib model [21], etc.). Amongst them, the so called cellular-level models [22], that is, SSN based, account for the most detailed approach as they capture the biophysical features of cerebellar neuronal processing. These models enable a realistic implementation of the current biological understanding of the cerebellum. Our cerebellar controller belongs to this cellular-level family and its very nature allows taking advantage of previous neurocomputational studies, which deepen in diverse aspects of cerebellar structure and functionality. For instance, how spike-timing-dependent plasticity (STDP) mechanisms can correlate the actual and desired motor states toward the generation of accurate corrective commands, even in the presence of sensorimotor delays [23]; synaptic adaptation at the cerebellar cortex: at the granular [24], [25] and molecular layer [26]; the suitability of cerebellar control under dynamic and kinematic perturbations [27]–[29]; the existence of structural and functional cerebellar microcomplexes [30]; the effective representation of neural input states for a supervised-learning cerebellar network [31]; or the granular layer timing mechanisms [32], information processing [33], and multimodal sensory inputs [34].

The depicted scenario yields several elements: 1) the cerebellum; a highly regular neural structure, thus, easy to computationally replicate to some extent, which is responsible for motor learning and coordination; 2) an artificial SNN incorporating a continuous learning process at its core that is able to mimic biological neurons and neural processing; and 3) hardware compliant robots lacking compliant control strategies. Here, we conjugate these three elements taking a holistic approach in tackling the physical HRI compliance problem.

Addressing this problem implies the state-of-the-art challenges that we face along this article.

First, we need the cerebellar-like SNN to operate in real time (RT). Spiking neural processing in RT is a highly demanding computational task. Considering that our computational resources are limited, there must be a tradeoff between network size, neuron complexity, network topology, and temporal output resolution, which determines, to a certain extent, the motor control accuracy. We further developed our spiking neural simulator (EDLUT) to accommodate, for the first time, an RT cerebellar SNN consisting of ~62K leaky integrated and fired (LIF) neurons with ~36.4M synapses, 36M of which are endowed with plasticity.

Second, we need to implement an effective RT dialog between the network spike domain and sensorimotor analog

![Diagram of Cerebral–cerebellar and spine–cerebellar anatomic connectivity](image-url)
that inherently allows for safe, compliant physical HRI. These SEAs interpose a spring between the motor/gearing elements and the final motor output. These springs are deformable under human interaction and, therefore, a built-in mechanism that inherently allows for safe, compliant physical HRI.

The Baxter robot, manufactured by Rethink Robotics [39], is a collaborative robot consisting of two arms with seven DoF. Baxter implements torque control and it is inherently compliant thanks to its series elastic actuators (SEAs) [40]. These SEAs interpose a spring between the motor/gearing elements and the final motor output. These springs are deformable under human interaction and, therefore, a built-in mechanism that inherently allows for safe, compliant physical HRI.

Prior to Baxter’s hands-on testing, we used the simulated version of Baxter available in Gazebo as a safe environment to develop and test the robot–cerebellum interface [41]. This interface was developed using ROS to control both the simulated and real robot. ROS allowed sending motor commands (torque commands) to the robot and receiving sensorimotor information (joints positions and velocities) from the robot sensors [42]. The designed motor tasks for our study involved the torque control of 6-DoF of Baxter’s left arm.

The Baxter robot, manufactured by Rethink Robotics [39], is a collaborative robot consisting of two arms with seven DoF. Baxter implements torque control and it is inherently compliant thanks to its series elastic actuators (SEAs) [40]. These SEAs interpose a spring between the motor/gearing elements and the final motor output. These springs are deformable under human interaction and, therefore, a built-in mechanism that inherently allows for safe, compliant physical HRI.

Prior to Baxter’s hands-on testing, we used the simulated version of Baxter available in Gazebo as a safe environment to develop and test the robot–cerebellum interface [41]. This interface was developed using ROS to control both the simulated and real robot. ROS allowed sending motor commands (torque commands) to the robot and receiving sensorimotor information (joints positions and velocities) from the robot sensors [42]. The designed motor tasks for our study involved the torque control of 6-DoF of Baxter’s left arm.

B. Cerebellar Control Loop

In order for the Baxter robot and the cerebellar network to exchange sensorimotor information, a dialog needed to be established. This dialog was framed within a closed control loop with negative feedback. See Fig. 2 for a control-loop overview.

The cerebellar-like spiking model (implemented in EDLUT, see below) acted as the controller and computed a motor command at each time step (2 ms) to achieve the goal behavior. To this aim, the controller computed the neural activity using the input information of the robot state, the ideal trajectory to be performed by the robot arm, and the instructive signal. The robot state (actual position, $Q_a$, and actual velocity, $\dot{Q}_a$, per joint) was provided by Baxter’s sensors and then mapped into control signals. The desired trajectory signals (position, $Q_d$, and velocity, $\dot{Q}_d$, per joint) were provided by a trajectory generator module representing the motor cortex and other motor areas [43]–[45]. The instructive $\varepsilon$ signals (one per joint) were obtained by comparison of the desired trajectory and the robot state signals. Once the cerebellar network computed a motor command, it was sent to the robot inducing movement to the arm. Consequently, the cerebellar network input sensory information was modified, thus, closing the loop. Cerebellar input and output signals were updated every 2 ms (500 Hz), guaranteeing low latency and continuous communication, a mandatory requirement for RT control.

The cerebellar model ran in the SNN simulator EDLUT [46]–[48]. EDLUT has been specially designed to perform embodiment experimentation so that neural computation can be slowed down/speeded up to cope with RT requirements imposed by a real body, for example, humanoid robot [35], [36].

C. Cerebellar Controller—The Neural Network

The cerebellar network controller consisted of five neural layers: 1) mossy fibers (MFs); 2) granule cells (GCs); 3) climbing fibers (CFs); 4) Purkinje cells (PCs); and 5) deep cerebellar nuclei (DCN) (see Fig. 3). The cerebellar network was in turn divided into six microcomplexes [30], each one focusing on controlling a different Baxter’s joint.

The MFs constituted the input layer through which the input sensorimotor information (actual and desired joint position and velocity trajectories translated into spiking patterns) was conveyed toward the inner cerebellar network layers. These MFs projected excitatory afferents on both GCs and DCN. GCs, then, processed and recoded this sensorimotor information in a sparse somatosensory neural activity that was
later propagated by the parallel fibers (PFs) (i.e., excitatory GCs’ axons) to the PCs. These PCs, in turn, correlated this somatosensory activity coming from PFs with the neural activity conveyed by the CFs (i.e., excitatory inferior olive, IO, and axons). The CF neural activity, generated in the olivary system, represented the mismatch between the actual and desired trajectories per Baxter’s joint and acted as an instructive signal. PCs underwent synaptic plasticity, that is, a supervised mechanism that correlated both PF and CF neural activities and adapted the PFs synaptic weight distribution accordingly. The cerebellar input–output response was adjusted and, therefore, the error movement minimized [31] in subsequent executions. Finally, the DCN closed the cerebellar loop via the excitatory synapses coming from MFs and CFs together with the inhibitory synapses from PCs.

The DCN neural activity of each microcomplex ultimately drove each Baxter’s joint by means of a spike-to-torque command transformation.

1) MFs (240) were modeled as input fibers able to propagate the sensorimotor information toward GCs and DCN at each simulation time step (2 ms). These 240 fibers were organized into six groups of 40 fibers each, that is, one group per joint. Each MF group was in turn subdivided into four equal subgroups on which actual and desired joint positions and velocities were directly mapped. Only four nonoverlapped MFs per group were active at each simulation time step representing the actual input neural state.

2) GCs (60,000) were modeled as LIF neurons emulating a state generator [26], [32], [33]. These 60,000 neurons were organized into six groups of 10,000 neurons each, that is, one group per joint. Each GC received four input synapses [34] coming from each subgroup belonging to the very same MF group. The connectivity pattern between MF and GC groups was designed in a way that nonoverlapped GC neural activation could univocally represent all possible MF neural input combinations. The granular and molecular cerebellar layers operating as a state generator transformed the sensorimotor neural information that each GC received from both external sources (via MFs) and other interneurons (via Golgi cells, Lugaro cells, unipolar brush cells, etc.) into nonoverlapped spatiotemporal patterns of neural activations. The granular layer, then, acted as a reservoir of interacting spiking neurons within a recurrent topology, whereas the subsequent PC acted as a readout layer. This state generator assumption allowed us merging granular and molecular cerebellar layers into one “granular layer” (see [49] for further details). Importantly, the MF-GC connectivity pattern facilitated the generation of these spatiotemporal neural states and their later readout.

3) CFs (600) were modeled as input fibers able to propagate the instructive signal (mismatch between the actual and desired trajectories per joint) toward PCs and DCN. These 600 fibers were organized into six microcomplexes of 100 neurons each, that is, one per joint. Each microcomplex was also divided into two symmetrical subgroups, each one dedicated to controlling the clock/anticlockwise movement of the robot joint actuator (emulating the agonist/antagonist interplay in human muscles). A probabilistic Poisson process transformed the error obtained when comparing the actual and desired trajectories per joint into CF spiking neural activations. Each CF spike encoded well-timed information regarding the instantaneous error. The probabilistic spike sampling of the error ensured a proper representation of the entire error region over trials, whilst maintaining the CF activity between 1 and 10 Hz per fiber (similar to electrophysiological data [50]). The error evolution could be sampled accurately even at such a low frequency [23], [51].

4) PCs (600) were modeled as LIF neurons. These 600 neurons were organized into six microcomplexes of 100 neurons each, that is, one per joint. Each microcomplex was also divided into two symmetrical subgroups, each one dedicated to controlling the clock/anticlockwise movement of the robot joint actuator. Each PC was connected to all PFs, thus receiving the sensorimotor information concerning all joints at once. CFs and PCs were one-to-one connected maintaining the six microcomplex architecture. Thus, each PC microcomplex received the same sensorimotor information via PFs, but a different instructive signal through its corresponding CFs microcomplex. Correlating these two different sources of neural information allows each PC microcomplex to adapt the cerebellar input-to-output
response of each Baxter’s joint via a plasticity mechanism that modified the overall PF synaptic weight distribution (see synaptic weight section).

5) DCN (600) were modeled as LIF neurons. These 600 neurons were organized into six microcomplexes of 100 cells neurons each, that is, one per joint. Each microcomplex was also divided into two symmetric subgroups, each one dedicated to controlling the clock/anticlockwise movement of the robot joint actuator. Each DCN cell was innervated by an inhibitory afferent from a PC and an excitatory afferent from the CF which simultaneously innervated the same PC. Each DCN cell also received excitatory projections from all MFs, which maintained the baseline DCN activity. This neural topology has been summarized in Table I.

Neuron and synapse numbers were limited by the computing resources available; however, the connectivity ratios (divergence and convergence ratios) between cerebellar layers were preserved according to MF-GrC, GrC-PC [52], MF-DCN, PC-DCN, CF-PC, and CF-DCN [53]–[55]. PC-DCN smaller convergence ratio was compensated by using higher synaptic weight values, thus maintaining the input drive.

The DCN neural activity was then transformed into an analog torque command ($T$) per microcomplex and then sent to Baxter’s actuators. This spike to analog conversion was computed at each time step, $T_{step} = 0.002$ s, using

$$DCN_{j,t} = \sum_{i=1}^{15} DCN_{j,i}(t)$$

$$DCN_j(t) = \sum_{i=1}^{N=100} DCN_{j,i}(t) - \sum_{i=51}^{N=50} DCN_{j,i}(t)$$

$$\tau_j(t) = \frac{\alpha_j}{15} \cdot \sum_{x=1}^{15} DCN_{output}(t-(x-1)\times T_{step})$$

where $j \in \{1, 6\}$ stands for the number of Baxter’s joints; $i \in \{1, 100\}$ defines the DCN tag number within the microcomplex related to joint $j$ (first 50 DCN cells encoding the agonist movement, last 50 DCN cells encoding the antagonist movement); and $\delta(t)$ stands for the Dirac delta function representing a spike event.

The spike to analog conversion in (1) and (2) was then convolved with a mean filter (3) acting as a DCN activity eligibility trace; that is, a temporary record of the occurrence of DCN previous spike events. The 15-taps mean filter helped us to emulate the low-pass filter behavior of muscles. The final torque output per joint was finally modulated by a factor $\alpha$ to adequate the normalized DCN output to the joint relative position, orientation, and mass; $\alpha_j = (0.75, 1.0, 0.375, 0.5, 0.05, 0.05)$ N-m/spike.

### D. Spiking Neuron Models

The cerebellar neural network consisted of LIF neurons [56] due to their minimal computational cost in spike generation and processing, a key factor in RT computation. Our LIF neurons only elicited a spike once their corresponding membrane potential reached a certain threshold and, immediately after, their membrane potentials were reset. The LIF neural dynamics was just defined by its membrane potential and its excitatory (AMPA and NMDA) and inhibitory (GABA) chemical conductances as follows:

$$C_m \cdot \frac{dV}{dt} = I_{internal} + I_{external}$$

(4)

$$I_{internal} = -g_l \cdot (V + E_L)$$

(5)

$$I_{external} = -(g_{AMPA}(t) + g_{NMDA}(t) \cdot g_{NMDA-INF}) \cdot (V - E_{AMPA})$$

$$- g_{GABA}(t) \cdot (V - E_{GABA})$$

(6)

$$g_{AMPA}(t) = g_{AMPA}(0) \cdot e^{-t \cdot \tau_{AMPA}} + \sum_{i=1}^{N} \delta_{AMPA}(t) \cdot w_i$$

(7)

$$g_{NMDA}(t) = g_{NMDA}(0) \cdot e^{-t \cdot \tau_{NMDA}} + \sum_{i=1}^{N} \delta_{NMDA}(t) \cdot w_i$$

(8)

$$g_{GABA}(t) = g_{GABA}(0) \cdot e^{-t \cdot \tau_{GABA}} + \sum_{i=1}^{N} \delta_{GABA}(t) \cdot w_i$$

(9)

$$g_{NMDA-INF} = \frac{1}{1 + \exp(62 \cdot V)} \cdot \frac{1.2}{1.5}$$

(10)

where $C_m$ denotes the membrane capacitance; $V$ is the membrane potential; $I_{internal}$ is the internal current; and $I_{external}$ is the external current. $E_L$ is the resting potential and $g_L$ is the conductance responsible for the passive decay term toward the resting potential. Conductances $g_{AMPA}$, $g_{NMDA}$, and $g_{GABA}$ integrate all the contributions received by each receptor type (AMPA, NMDA, and GABA) through individual synapses, being $g_{NMDA-INF}$ as the NMDA activation channel. These conductances were defined as decaying exponential functions [46], [56] where their values were directly

### Table I: Neural Network Topology

| Neurons | Synapses                  | Number | Type   | Weight (nS) | Range (nS) |
|---------|---------------------------|--------|--------|-------------|------------|
| Pre-synaptic cells | Post-synaptic cells | 240 MFs | 60 K GCs | 2400 | AMPA | 0.18 | - |
| 240 MFs | 600 DCN | 144K AMPA | 0.1 | - |
| 60 K GCs | 600 DCN | 36 M AMPA | 1.6 | [0, 5] |
| 600 PCs | 600 DCN | 600 AMPA | 1.0 | - |
| 600 CFs | 600 PCs | 600 AMPA | 0.25 | - |
| 600 CFs | 600 DCN | 600 AMPA | 0.1 | - |

### Table II: Neurons Model Parameters

| Parameters | GC | PC | DCN |
|-----------|----|----|-----|
| $C_m$ (pf) | 2.0 | 100 | 2.0 |
| $g_L$ (nS) | 1.0 | 6.0 | 0.2 |
| $E_L$ (mV) | -65.0 | -70 | -70.0 |
| $E_{AMPA}$ (mV) | 0.0 | 0.0 | 0.0 |
| $E_{GABA}$ (mV) | -80.0 | -80.0 | -80.0 |
| $\tau_{AMPA}$ (ms) | 1.0 | 1.2 | 0.5 |
| $\tau_{NMDA}$ (ms) | 14.0 | 14.0 | 14.0 |
| $\tau_{GABA}$ (ms) | 10.0 | 10.0 | 10.0 |
| $V_m$ (mV) | -50.0 | -52.0 | -40.0 |
| $T_{step}$ (ms) | 1.0 | 2.0 | 1.0 |
incremented proportionally to the synaptic weights \(w_i\) upon each presynaptic spike arrival (Dirac delta functions). When the membrane potential reached a threshold \(V_{th}\), it was then reset to \(E_L\) during the refractory period \(T_{ref}\). The configuration parameters for the three neurons modeled are shown in Table II.

**E. Synaptic Plasticity**

The adaptive motor process of the cerebellar network was implemented through an STDP mechanism located at PF-PC synapses. This STDP mechanism balanced long-term potentiation (LTP) and long-term depression (LTD) at PC synaptic level as follows:

\[
\text{LTP} \Delta w_{PFj-PCi}(t) = \alpha \cdot \delta_{PFspike}(t) \cdot dt \tag{11}
\]

\[
\text{LTD} \Delta w_{PFj-PCi}(t) = \beta \cdot \int_{-\infty}^{t_{CFspike}} k(t - t_{CFspike}) \cdot \delta_{PFspike}(t) \cdot dt \tag{12}
\]

where \(\Delta W_{PFj-PCi}(t)\) denotes the synaptic weight change between the \(j\)th PF and the target \(i\)th PC; \(\alpha = 0.002\) nS is the synaptic efficacy increment; \(\delta_{PF}\) is the Dirac delta function corresponding to an afferent spike from a PF; \(\beta = -0.001\) nS is the synaptic efficacy decrement; and the kernel function \(k(x)\) is defined as

\[
k(x) = \begin{cases} 
- \frac{x + \delta_k}{\tau_{LTD} - \delta_k} \cdot e^{\frac{x + \delta_k}{\tau_{LTD} - \delta_k}} + 1 & \text{if } x < -\delta_k \\
0 & \text{if } x \geq -\delta_k 
\end{cases} \tag{13}
\]

where \(\tau_{LTD} = 100\) ms is the time constant that is aligned with the biological sensorimotor pathway delay (~100 ms), the time period elapsed from the sensory information reception to information transmission along nerve fibers, neural processing time responses, and the final motor output response [57]. \(\delta_k = 0.07\) s allows for the adjustment of the kernel width. The kernel maximum value \((k(x) = 1)\) is obtained when \(x = -\tau_{LTD}\), and zero or close to zero when \(x > -\delta_k\) or \(x < -\tau_{LTD} - 10 \cdot (\tau_{LTD} - \delta_k)\). The STDP rule defined by (11)–(13) caused a fixed synaptic efficacy increment (LTP) each time a spike arrived through the PFs to the target PC and a variable synaptic efficacy decrement (LTD) each time a spike arrived through a CF to the target PC. The amount of synaptic decrement depended on the activity arrived through the PFs prior to the CF spike. Both activities were convolved using the integrative kernel defined in (13) and were multiplied by the synaptic decrement \(\beta\). The effect on the presynaptic spikes arriving through PFs was maximal during the 100 ms time window \((\tau_{LTD} = 100\) ms) before the postsynaptic CF spike arrival, thus accounting for the sensorimotor pathway delay [23], [27], [58].

This STDP mechanism correlated the neural activity patterns coming through the PFs toward PCs with the instructive signals coming from CFs toward PCs. This correlation process at the PC level identified certain PF activity patterns codifying certain sensorimotor information and, consequently, diminished the PC output activity by a PF-PC synaptic weight reduction. A reduction on the PC activation caused a subsequent reduction on the PC inhibitory action over the target DCN. Conversely, in the absence of any correlation, the STDP mechanism increased the PC output activity by a PF-PC synaptic weight potentiation. Since the DCN were driven by a near constant baseline MF activation, a lack of PC inhibitory action would cause an increasing DCN activity whereas an incremental PC inhibitory action would do otherwise. Well-timed sequences of increasing/decreasing levels of DCN activation during the learning acquisition process ultimately shaped the cerebellar output activity and diminished the overall error.

**F. ROS Modules Implementation**

The control loop consisted of three main elements: 1) trajectory generator; 2) cerebellar controller; and 3) Baxter robot. The implementation and communication amongst these three elements were developed using ROS, allowing modularity. Fig. 4 depicts the control-loop diagram. Each block defines an ROS module and each black arrow represents an ROS topic that establishes the communication between ROS modules exchanging either analog signals or spike trains.

This control loop was designed accounting for the sensorimotor pathway delay (~100 ms) [59]. The 100 ms delay comprised the efferent \((\delta_e = 50\) ms) and afferent \((\delta_a = 50\) ms) pathway delays (Fig. 4 dashed red arrows). A motor command originated at time \(t\) on the cerebellum was applied by the robot actuators at time \(t + \delta_e\) and its effect sensed back at the cerebellar network at time \(t + \delta_e + \delta_a\). The cerebellar plasticity mechanism described in (11)–(13) compensated for this sensorimotor delay.

The control loop assisted the cerebellar controller in the generation of the torque commands able to minimize the mismatch between the reference signal (desired joint position and velocity) and the robot state (actual joint position and velocity). To that, different ROS modules were implemented.

1) The trajectory generator module generated the desired trajectory signals, whilst the Baxter robot generated the actual state signals and executed the motor commands.
2) The RT cerebellum module accommodated the cerebellar controller implemented in EDLUT (imported in ROS as a C++ library). This module received, computed, extracted, and propagated the neural activity toward the next ROS module.

3) Desired, actual, and instructive signals needed to be transformed into spike trains that the cerebellum could process. The MF and CF analog2spike modules carried out this transformation.

4) The error estimator module provided the cerebellar controller with the instructive signal needed for neural adaptation. The error estimator module required comparing desired and actual trajectories.

5) The cerebellar output spiking signals needed to be transformed into analog commands that the Baxter robot could process. The DCN spike2analog module, using (1) and (2) transformed the spike trains into torque commands, which were lately smoothed out by the mean filter module using (3).

6) The torque command module closed the loop sending the torque commands obtained from the mean filter module to the Baxter robot.

7) The supervisor module was implemented as a safe mechanism mimicking mechanical brakes. A supervisor module maintained Baxter within a safe working range during the first stages of neural adaptation. Only at the event of any of the joints getting outside its working range, the supervisor module added a corrective torque value to the cerebellar torque command to prevent damages.

All modules were synchronized thanks to a reference time signal extracted from Baxter’s internal clock running under the network time protocol (NTP), ensuring RT [60]. Each event, that is, analog signal or spike train, generated on an ROS module carried a time stamp indicating the event processing time to the subsequent module. Each target module incorporated an input buffer in which events were stored for later synchronous processing according to their timestamps. The RT cerebellum module, however, allowed asynchronous processing of the events stored at its input/output activity buffers thanks to the RT mechanism incorporated in EDLUT [35]. On the event of empty input buffers, the neural simulation was halted. On the event of an almost empty output buffer, the neural simulation was speeded up (see [35] for an in-depth review on RT neural simulation). Hence, the RT cerebellum module could deal with neural activity volleys encountered during the cerebellar simulation that could not be processed synchronously.

G. Benchmarking the Cerebellar Controller—Behavioral Tasks

We drew inspiration from the cerebellar role in motor control and movement coordination [15], [17] to implement a novel control strategy for hardware compliant robots. It is thus appropriate to evaluate our cerebellar-like model in the field of robot dynamics control in terms of performance under a set of different conditions. To this aim, we proposed a specific way of performing the experimental evaluation through two trajectory families.

1) On the one hand, we tested our cerebellar controller in reaching movements; that is, fast, ballistic arm movements with bell-shaped velocity profiles, that is, s-curve, toward a target point [61]. Arm reaching movements are primarily used for characterizing cerebellar pathologies in human motor control by measuring the time to target and precision to target. Arm dynamics control is critical due to the constraint at stake when moving masses. A single-joint limb movement in fast de/acceleration causes motion in all other limb joints thus arising interaction forces to be compensated by the cerebellum as well as our controller [62].

2) On the other hand, we tested our cerebellar controller facing a set of fast movements in smooth trajectories consisting of sinusoidal-like profiles for both position and velocity per joint. The end-effector shall describe either circular or eight-like Cartesian trajectories in the horizontal plane [29], [63]. These trajectories are well suited for revealing the complex dynamics of a 6-DoF robotic arm [64], including interaction forces to be compensated by the cerebellar controller [62].

These trajectory families were first designed in the Cartesian space, providing 3-D position and orientation for the end-effector, and then translated into joint space using MoveIt! software [65]. This offline process allowed the precomputation of joint space trajectories (position and velocity per joint) that were later on used by the trajectory generator module to build the desired trajectory signals, used as cerebellar input.

The circular trajectory in the Cartesian space meets (14), whilst (15) describes the eight-like trajectory

\[
\begin{align*}
\begin{cases}
  x = R \cdot \cos(2 \cdot \pi \cdot t \cdot 0.5) \\
  y = R \cdot \sin(2 \cdot \pi \cdot t \cdot 0.5) \\
  z = \alpha \\
  x = 0.5 \cdot R \cdot \sin(2 \cdot \pi \cdot t \cdot 0.5) \\
  y = R \cdot \cos(2 \cdot \pi \cdot t \cdot 0.5) \\
  z = \alpha
\end{cases}
\end{align*}
\]

where \( R \) denotes a 120 mm radius which is halved for the \( x \) coordinate in (15) to keep the eight-like trajectory within the working space limits of the robot. The \( z = \alpha \) coordinate and the end-effector vertical orientation were kept constant to maintain the horizontal plane through the trajectories. Each trajectory lasted 2 s.

Once the translation from Cartesian \((x, y, z)\) to joint space positions \((Q_1 \text{ to } 6)\) was completed, the joint velocity profiles \((Q_1 \text{ to } 6)\) were obtained as the position derivative over time.

Regarding the target-reaching task, the center of the circle trajectory was the starting position. Eight different points along the circular trajectory perimeter constituted the reaching targets following an even distribution at every \( \pi/4 \) radians. As aforementioned, this task tested the controller through point-to-point multijoint movements with s-curve velocity profiles that provided fast acceleration/deceleration changes, that is, ballistic movements. The subsequent high-jerk values entailed
high inertial forces to be compensated by the cerebellar controller. Each target-reaching movement lasted 2 s back and forth between the target and the central position, that is, 1 s to reach the target and 1 s to go back to the central position. These three different behavioral tasks provided us with a varying context to test the cerebellar network. For every task, the cerebellar network acquired those motor commands needed to achieve the desired goal behavior through learning. The learning process was accomplished through the repetition over time of a specified trajectory.

The performance evaluation was carried out comparing the goal and the actual behavior, that is, the desired and the actual joint positions. The average difference constitutes the position mean absolute error (MAE), which is our performance evaluation metric following the below equations:

\[
\text{MAE}_{\text{joint}} = \sum_{i=0}^{K} (Q_i, \text{desired} - Q_i, \text{actual})
\]

\[
\text{MAE} = \frac{\sum_{j=1}^{N} \text{MAE}_j}{N}
\]

where \( K = 1000 \) denotes the number of samples of the two second trajectories; and \( N = 6 \) is the number of joints. The MAE provided a numerical performance indicator for the quality of the cerebellar controller, thus allowing us to compare it against the default factory-installed position control.

For reproducibility and comparative purposes, the experimental setup (benchmarking included), source code, and experimental results are available at: https://github.com/EduardoRosLab/EDLUT_BAXTER.

III. RESULTS

We tested our cerebellar-like controller in different behavioral tasks, considering the default factory-installed position control mechanism as a performance baseline to validate the results. The aforementioned circular, eight-like, and target reaching trajectories constituted our cerebellar benchmarking, which was completed with a set of interactions in an unstructured environment to test compliance.

A. Circular Trajectory

This first behavioral task consisted of following a 120-mm radius circular path in the horizontal plane (xy) repeated over time to facilitate learning and adaptation, each trial having a time duration of 2 s. The STDP mechanism governing the learning process modulated the cerebellar output (see methods), driving the robot’s behavior toward the goal. The behavioral evolution through time is illustrated in Fig. 5. Three snapshots were taken at three different moments of the cerebellar learning process: 1) initial stage; 2) intermediate stage; and 3) final stage.

1) Initial Learning Stage: The cerebellar-model started learning from scratch. At an initial learning stage [Fig. 5(left column)], the synaptic adaptation mechanism at PF-PC synapses that correlated the somatosensory information with the CF instructive signal was not effectively deployed yet. Thus, the inhibitory action from PCs onto DCN was of marginal utility; making the DCN output activity saturated as it solely responded to the excitation coming from MF and CF afferents.
Fig. 6. Behavioral evolution through eight-like trajectory trials (2 s). (a) Initial learning stage ($t_1 = 18$–$20$ s). (b) Intermediate learning stage ($t_2 = 318$–$320$ s). (c) Final learning stage ($t_3 = 998$–$1000$ s). The first row depicts the cerebellar output activity (DCN layer), whereas the second row shows its analog conversion into torque commands. The third row illustrates the desired versus actual trajectory per joint. (d)–(f) reveal the desired versus actual trajectory of the end-effector at $t_1$, $t_2$, and $t_3$, respectively. Also the density functions corresponding to the prior ten trials are depicted. (g) represents the position MAE per trial through the learning process. The MAE of each joint is illustrated as well as the average MAE of all joints, completed with the default factory-installed position control baseline performance.

As depicted in Fig. 5(d), the density function generated from ten trials before $t_1$ snapshot [Fig. 5(left column)] reveals that the robot was still exploring the working area, performing low consistent, dispersed movements.

2) Intermediate Learning Stage: At an intermediate learning stage [Fig. 5(central column)], the synaptic adaptation allowed the recognition of some somatosensory patterns at the PCs, reflected in an emerging differentiated DCN activity between agonist and antagonist subgroups at each microcomplex [Fig. 5(b), first row]. Consequently, the robot’s behavior began getting closer to the desired goal [Fig. 5(b), third row; and (e)].

3) Final Learning Stage: Once the learning process reached advanced stages [Fig. 5(right column)] the robot executed the desired trajectory with minimal error. The agonist/antagonist DCN activity was clearly differentiated at each microcomplex [Fig. 5(c), first row], and translated into the required torque commands via a spike-to-analog conversion (see methods). The synaptic adaptation process was reflected in a clear evolution of the torque values compared to previous stages, directly affecting the robot output behavior. All joints closely followed the desired trajectory at this stage [Fig. 5(c), third row] and, consequently, the end-effector barely missed at describing the desired circular path [Fig. 5(f)], having a consistent behavior around the goal trajectory over trials.

The overall performance through the learning process is depicted in Fig. 5(g); illustrating how the cerebellar-like controller performance was improved as adaptation and learning were fulfilled. MAE evolution indicates that the cerebellar controller needed about 300 trials (i.e., 600 s) to converge, outperforming the accuracy of the default factory-installed position control baseline (0.019 ± 0.003 versus 0.077 ± 0.0004, Table III).

### TABLE III

| CIRCULAR AND EIGHT-LIKE TRAJECTORIES: LEARNING STAGES MAE |
|----------------------------------------------------------|
| Cerebellar torque control (trials) | Position control (trials) |
|-----------------------------------|--------------------------|
| [0-100]                           | [0-100] |
| 0.115 ± 0.036 ± 0.019 ± 0.055 ± 0.013 ± 0.003 ± 0.077 ± 0.004 | 0.063 ± 0.003 |
| [100-200]                         | [100-200] |
| 0.034 ± 0.046 ± 0.017 ± 0.034 ± 0.013 ± 0.003 ± 0.004 |
| [400-500]                         | [400-500] |
| [0.500]                           | [0.500] |

B. Eight-Like Trajectory

The eight-like trajectory was concentric to the previously discussed circle shaped; it had a “radius” of 120 mm and each trial lasted 2 s. In terms of robot dynamics, the eight-like trajectory was more demanding than the circular trajectory; as faster and steeper changes in velocity module and direction were required for trajectory completion [64]. Benchmarking control capacities were further increased by allowing the linear and angular velocity of the Baxter’s end-effector to vary within...
Fig. 7. Behavioral evolution through target reaching trials (2 s). Each trial consisted of one of the eight possible tasks. (a) Initial learning stage \( (t_1 = 158–160 \text{ s}) \), (b) Intermediate learning stage \( (t_2 = 598–600 \text{ s}) \), (c) Final learning stage \( (t_3 = 1998–2000 \text{ s}) \). (a)–(c) depict the last performed trajectory for each of the eight possibilities in the Cartesian space prior to \( t_1 \), \( t_2 \), and \( t_3 \), respectively. The density functions reveal the end-effector behavior over the last 80 trials, grouping the eight possible tasks by trajectory direction. (d)–(f) show the velocity profiles related to the target reaching trajectory marked with * in the Cartesian space. The illustrated trials correspond to the last * iteration prior to \( t_1 \), \( t_2 \), and \( t_3 \), respectively. Note that the rising and lowering times achieved are consistent with human data (~250–500 versus ~400–500 ms) in 1 s target reaching movements [72]. (g) represents the position MAE per trial through the learning process. The MAE of each joint is illustrated as well as the mean MAE of all joints. High standard deviation values reflect how some reaching movements were more demanding than others. The position control baseline is the average MAE of the default factory-installed under the same stochastic distribution over trials.

Fig. 8. Performance in an unstructured environment. Whilst performing the already learned circular trajectory, a set of unstructured interactions was undertaken: 1) a 1/2 kg payload was attached to the end-effector and later on detached; 2) an elastic band was attached to the end-effector and later on detached; and 3) a series of physical HRIs. The figure depicts the position of MAE through trials as interactions is undertaken, illustrating the cerebellar adaptation to unknown scenarios.

C. Target Reaching

This task consisted of eight different reaching movements sharing the same starting point. The challenge lied in the high speed of the movements and the randomness in the order of trials (transitions between the eight reaching movements were stochastic). The growth in complexity for the cerebellar controller was illustrated by a lower MAE convergence speed entailing higher standard deviation values inter trials and the need of more trials to reach stability than in the two previous behavioral tasks (Table IV). Nevertheless, the cerebellar-like controller was able to perform these ballistic movements, improving its performance through learning and reaching again better accuracy than the default factory-installed position control mechanism [Fig. 7] (0.017 ± 0.003 versus 0.026 ± 0.006).

Therefore, not only the cerebellar-like controller was able to perform accurate smooth trajectories but also fast-ballistic movements.

the trajectory [63], [66]–[68] (see the Appendix). Nonetheless, the obtained results were equally satisfying (see Table III).

1) Initial Learning Stage: At an early learning stage [Fig. 6(left column)], the robot’s behavior was clearly far from the desired goal. DCN activity at this stage responded exclusively to the excitatory drive from MF-DCN and CF-DCN afferents, thus, it was saturated [Fig. 6(a), first row]. The MAE value was high (0.165) and the performed trajectory was far from the goal [Fig. 6(a), third row; (d), and (g)].

2) As learning progressed, the PF-PC synaptic adaptation mechanism began shaping the DCN activity causing an incipient neural activity differentiation between agonist and antagonist microcomplexes [Fig. 6(b), first row]. In consequence, the corresponding torque values significantly differed from those of early stages [Fig. 6(b), second row], and the robot’s behavior began getting closer to the desired one [Fig. 6(b), third row; and (e)].

3) Finally, once learning was fully deployed the robot behaved as desired [Fig. 6 (c), third row; and (f)]. The DCN activity was clearly sculpted to produce the needed torque commands to perform the desired trajectory [Fig. 6(c)], maintaining a stable behavior over trials (0.017 ± 0.003).

The greater difficulty of the eight-like trajectory was noted in a lower convergence speed for the cerebellar-like controller to reach a stable behavior. (Table III shows a slower MAE convergence speed than the circular trajectory.) However, the final performance accuracy obtained also outperformed the default factory-installed position control baseline (0.017 ± 0.003 versus 0.063 ± 0.0003).
TABLE IV

| MAE | Cerebellar torque control (trials) | Position control (trials) |
|-----|----------------------------------|--------------------------|
|     | [0-100] | [300-400] | [900-1000] | [0-1000] |
|     | 0.155 ± 0.043 ± 0.019 ± 0.059 | 0.024 ± 0.006 |

D. Unstructured Interactions

Aiming at testing the compliance of the cerebellar controller, we tested its response in an unstructured environment. Whilst performing the circular trajectory, some interactions were undertaken [Fig. 8]. First, the dynamics of the robotic arm was modified in two different ways.

1) By adding a 0.5 kg payload to the end-effector attached to a rod, mimicking a pseudo “conical pendulum.” The tension force of the rod acting on the robot varied with the alignment between the payload and the end-effector.

2) By attaching an elastic band to apply an elastic force that tried to return the band to its natural length.

In both cases, the cerebellar-like controller successfully adapted to the new context after a learning period.

Subsequently, human interactions were performed:

1) a human was able to move the robotic arm by applying an extremely low force (i.e., one-finger push);
2) a human grabbed the robotic arm and moved it through the working space with no opposition from the robot;
3) a human got in the way of the robotic arm trajectory with no risk of injury.

These results allow us to confirm that the cerebellar-like controller was able to accurately perform the desired trajectories, no matter the dynamics modifications; and guaranteed a safe HRI as no damages were suffered when interrupting the robot’s task, at either human or robot side.

Four movies are included as supplementary material to fully illustrate the cerebellar learning and adaptation process. The target reaching, eight-like, and circular trajectory movies show from up to down and left to right the following clips, all of them playing synchronized RT information: 1) a frontal shot of the robot performing the trajectory; 2) the evolution of the position MAE per trial; 3) a nadir shot of the robot performing the trajectory; 4) the trajectory being performed by the end-effector in the Cartesian space; 5) the cerebellar output activity (DCN layer spikes); and 6) the corresponding torque commands obtained from the spike-to-analog conversion of the DCN activity. Different cuts corresponding to an initial, intermediate, and final learning stage verify the behavioral evolution.

Finally, the unstructured environment movie shows the cerebellar adaptation and, therefore, robot adaptation, to unknown, unstructured scenarios; thus, proving compliance.

IV. DISCUSSION

The role of the cerebellum in controlling human motor coordination has led and crystallized into different cerebellar modeling approaches in the robotics control field. According to their understanding of the cerebellar operation, these models can be categorized as follows [22].

1) Functional Models: A functional understanding of specific cerebellar operations suffices to build these nonbiological models. By setting aside any attempt to compel biological accuracy, these models offer the most computationally friendly approach to provide motor learning and control. The MPFIM [20] model follows this approximation. Functional models can be found in [69].

2) State-Encoder-Driven Models: These models operate as an abstraction of the granular and molecular layer of the cerebellum. They split up the state space assuming granule cells as on/off entities performing a mapping of inputs onto binary outputs. The CMAC [18], cerebellar adaptive filter [70], LWPR [71], or APG [19] models fall into this nonbiologically plausible category, used in [72]–[78].

3) Cellular-Level Models: A set of differential equations models each cerebellar neuron, simulating its biological behavior. These biological plausible models offer an insight into the cerebellar function on the cellular level. The Schweighofer–Arbib model [21] is an example of this category. The computational cost of these models usually limits them to small-sized networks [35], non-RT simulated scenarios [49], [79], or low-resolution output control signals [80].

Our SNN cerebellar controller fits in the cellular-level context. It stands out for its capacity to perform compliant robot control in RT, a feature that, to the best of our knowledge, was not accomplished before by means of a cerebellar SNN. The ~62K neurons and 36M synapses endowed with plasticity enable a detailed mapping of the sensorimotor space and the generation of precise output motor commands to achieve the goal behavior. Baxter’s factory-installed controller was set as the baseline reference with the sole function of validating our controller performance.

V. CONCLUSION

Physical HRI implies controlling nonlinearities at the robotic end, thus demanding adaptive control. In this article, taking inspiration from biology, we expand the family of RT adaptive robot controllers beyond machine learning [81],

Fig. 9. Baxter’s end-effector position and linear velocity when performing the eight-like trajectory. (a) and (b) Desired versus actual position and linear velocity of the end-effector. Trajectory points, \( P_1 \), \( P_2 \), \( P_3 \), and \( P_4 \), illustrate the relation high trajectory curvature–low velocity \( (P_2, P_4, P_6, \text{and} P_8) \) and low trajectory curvature–high velocity \( (P_1, P_3, P_5, \text{and} P_7) \) found in human hand curved motions [72]. Both graphs correspond to the end of the learning process (998–1000 s). Our adaptation process grants greater influence to position error when generating the instructive signal causing the velocity error to remain larger.
fuzzy logic [82], [83], and ANNs [9], [84] solutions. The intrinsic characteristics of SNNs, that is, timing codification of evolving sensorimotor states, make them an appealing approach for motor control architectures [11], [12]. We present a novel biologically plausible motor control architecture with a cerebellar-like SNN controller at its core that is able to drive a 6-DoF robot via torque commands in RT.

The implementation of a controller equipped with cerebellar plasticity mechanism (STDP) turns dispensable the availability of a detailed dynamic model of the robot. The cerebellar-like SNN is able to self-adapt and learn from scratch to control a given robot, making unnecessary any prior dynamics knowledge. Thus, the complexity of modeling nonlinear systems is tackled, and this SNN controller constitutes a plausible solution to control not only our Baxter robot but also any torque-controlled robot. Previously achieved SNN position control [80], [85] does not provide compliance as physical perturbations/interactions are not supported; hence, the importance of reliable torque control toward achieving safe physical HRI.

The variety of demanding behavioral tasks in terms of control requirements here accomplished proves our SNN cerebellar-like controller a valid solution. Our SNN controller succeeded in terms of position accuracy, high-speed movements, and compliance since the baseline performance (i.e., default factory-installed position control) was utterly improved in all the experimental behavioral tasks.

The development of biologically plausible controllers appears as a driving force for the evolution of robotics toward more advanced, intelligent, bioinspired, and compliant robots. Furthermore, the embodiment of biologically accurate ANNs implies a great opportunity for neuroscience studies. These neural network models can be computationally simulated under different biologically relevant tasks to give a consistent idea about how the CNS neural network may operate.

APPENDIX

Humans performing curved-profile hand motions show high linear velocities during segments of low curvature and low velocities during the segments of high curvature [86]. The eight-like trajectory consists of a combination of low/high curvature segments able to reveal the human performance in curved motions. We found that Baxter’s velocity profiles were biologically consistent, that is, the end-effector moved at high velocity when the trajectory curvature remained low and at low velocity when the trajectory curvature was high (see Fig. 9).

The eight-like velocity profile is given, ideally, by deriving the Cartesian trajectory (15) as in (18). The linear velocity of Baxter’s end-effector is finally given by (19)

\[
\begin{align*}
\dot{x} &= \pi \cdot R \cdot \cos(2 \cdot t \cdot \pi) \\
\dot{y} &= -\pi \cdot R \cdot \sin(2 \cdot t \cdot \pi) \\
\dot{z} &= 0
\end{align*}
\]

\[
v = \sqrt{\dot{x}^2 + \dot{y}^2 + \dot{z}^2}.
\]

ACKNOWLEDGMENT

The authors would like to thank Dr. F. Barranco and Dr. R. R. Carrillo for their research inputs.

REFERENCES

[1] M. A. Goodrich and A. C. Schultz, “Human–robot interaction: A survey,” Found. Trends Human Comput. Interact., vol. 1, no. 3, pp. 203–275, 2008.
[2] B. T. Volpe et al., “Robotic devices as therapeutic and diagnostic tools for stroke recovery,” Archives Neurol., vol. 66, no. 9, pp. 1086–1090, 2009.
[3] K. Wada and T. Shibata, “Living with seal robots—Its sociopsychological and physiological influences on the elderly at a care house,” IEEE Trans. Robot., vol. 23, no. 5, pp. 972–980, Oct. 2007.
[4] P. Baxter, E. Ashurst, R. Read, J. Kennedy, and T. Belpaeme, “Robot education peers in a situated primary school study: Personalisation promotes child learning,” PLoS ONE, vol. 12, no. 5, 2017, doi:10.1371/journal.pone.0178126.
[5] A. De Santis, B. Siciliano, A. De Luca, and A. Bicchi, “An atlas of physical human–robot interaction,” Mech. Mach. Theory, vol. 43, no. 3, pp. 253–270, 2008.
[6] A. Diamond, R. Knight, D. Devereux, and O. Holland, “Anthropomimetic robots: Concept, construction and modelling,” Int. J. Adv. Robotic Syst., vol. 9, no. 5, p. 209, 2012.
[7] D. Rus and M. T. Tolley, “Design, fabrication and control of soft robots,” Nature, vol. 521, no. 7553, p. 467–475, 2015.
[8] H. Chauoi, P. Sicard, and W. Gueaieb, “ANN-based adaptive control of robotic manipulators with friction and joint elasticity,” IEEE Trans. Ind. Electron., vol. 56, no. 8, pp. 3174–3187, Aug. 2009.
[9] R. M. Robinson, C. S. Kothera, R. M. Sanner, and N. M. Wereley, “Nonlinear control of robotic manipulators driven by pneumatic artificial muscles,” IEEE/ASME Trans. Mechatron., vol. 21, no. 1, pp. 55–68, Feb. 2016.
[10] S. Ghosh-Dastidar and H. Adeli, “Third generation neural networks: Spiking neural networks,” in Advances in Computational Intelligence. Berlin, Germany: Springer-Verlag, 2009, pp. 167–178.
[11] W. Maass, “Networks of spiking neurons: The third generation of neural network models,” Neural Netw., vol. 10, no. 9, pp. 1659–1671, 1997.
[12] D. V. Buonomano and M. D. Mauk, “Neural network model of the cerebellum: Temporal discrimination and the timing of motor responses,” Neural Comput., vol. 6, no. 1, pp. 38–55, Jan. 1994.
[13] S. P. Swinnen et al., “Shared neural resources between left and right interlimb coordination skills: The neural substrate of abstract motor representations,” Neuroimage, vol. 49, no. 3, pp. 2570–2580, 2010.
[14] J. Voogd and M. Glickstein, “The anatomy of the cerebellum,” Trends Neurosci., vol. 21, no. 9, pp. 307–375, 1998.
[15] J. S. Albus, “A theory of cerebellar function,” Math. Biosci., vol. 10, nos. 1–2, pp. 25–61, 1971.
[16] M. Ito, The Cerebellum and Neural Control. New York, NY, USA: Raven Press, 1984.
[17] M. G. Paulin, “The role of the cerebellum in motor control and perception,” Brain Behav. Evol., vol. 41, no. 1, pp. 39–50, 1993.
[18] J. S. Albus, “A new approach to manipulator control: The cerebellar model articulation controller (CMAC),” J. Dyn. Syst. Meas. Control, vol. 97, no. 3, pp. 220–227, 1975.
[19] J. C. Houk, J. T. Buckingham, and A. G. Barto, “Models of the cerebellum and motor learning,” Behav. Brain Sci., vol. 19, no. 3, pp. 368–383, 1996.
[20] D. M. Wolpert and M. Kawato, “Multiple paired forward and inverse models for motor control,” Neural Netw., vol. 11, nos. 7–8, pp. 1317–1329, 1998.
[21] N. Schweighofer, “Computational models of the cerebellum in the adaptive control of movements,” Ph.D. dissertation, Univ. Southern California, Los Angeles, CA, USA, 1995. Accessed: Oct. 16, 2019. [Online]. Available: http://digitallibrary.usc.edu/cdm/compoundobject/collection/p15799coll20/id/622886/edc1
[22] M. A. Arbib, G. Metta, and P. van der Smagt, “NeuroRobotics: From vision to action,” in Springer Handbook of Robotics. Cham, Switzerland: Springer-Verlag, 2008, pp. 1453–1480.
[23] A. De Santis, B. Siciliano, A. De Luca, and A. Bicchi, “Cerebellar-like corrective model inference engine for manipulation tasks,” IEEE Trans. Syst., Man, Cybern. B, Cybern., vol. 41, no. 5, pp. 1299–1312, Oct. 2011.
[24] J. A. Garrido, E. Ashurst, and D’Angelo, “Spike timing regulation on the millisecond scale by distributed synaptic plasticity at the cerebellum input stage: A simulation study,” Front. Comput. Neurosci., vol. 7, p. 64, May 2013.
M. Quigley et al. G. A. Pratt and M. M. Williamson, “Series elastic actuators,” in N. P. Koenig and A. Howard, “Design and use paradigms for E. Ros, R. Carrillo, E. M. Ortigosa, B. Barbour, and R. Agís, “Event- and time-driven techniques using parallel CPU-GPU co-processing for spiking neural networks,” Front. Neurosci., vol. 11, p. 7, Feb. 2017.

N. R. Luque, J. A. Garrido, R. R. Carrillo, and E. Ros, “Distributed cerebellar motor learning: A spike-timing-dependent plasticity model,” Front. Comput. Neurosci., vol. 10, p. 17, Mar. 2016.

S. Kuroda, K. Yamamoto, H. Miyamoto, K. Doya, and M. Kawato, “Statistical characteristics of climbing fiber spikes necessary for efficient cerebellar learning,” Biol. Cybern., vol. 84, no. 3, pp. 183–192, 2001.

R. R. Carrillo, E. Ros, C. Boucheny, and J.-M. C. Olivier, “A real-time spiking cerebellum model for learning robot control,” Biosystems, vol. 94, nos. 1–2, pp. 18–27, 2008.

W.-K. Li, M. J. Hausknecht, P. Stone, and M. D. Mauskopf, “Using a million cell simulation of the cerebellum: Network scaling and task generality,” Neural Netw., vol. 47, pp. 95–102, Nov. 2013.

M. Uissiari and E. De Schutter, “The mysterious microcircuitry of the cerebellar nuclei,” J. Physiol., vol. 589, pp. 3441–3457, Jul. 2011.

J. F. Medina and M. D. Mauskopf, “Simulations of cerebellar motor learning: computational analysis of plasticity at the mossy fiber to deep nucleus synapse,” J. Neurosci., vol. 19, no. 16, pp. 7140–7151, 1999.

J. F. Medina and M. D. Mauskopf, “Computer simulation of cerebellar information processing,” Nat. Neurosci., vol. 3, p. 1205–1211, Nov. 2000.

W. Gerstner and W. M. Küster, Spiking Neural Models: Single Neurons, Populations, Plasticity, Cambridge, U.K.: Cambridge Univ. Press, 2002.

A. Sargolzaei, M. Abdelghani, K. Y. Ken, and S. Sargolzaei, “Sensorimotor control: Computing the immediate future from the delayed present,” BMC Bioinformatics, vol. 17, no. 7, p. 245, 2016.

M. Kawato and H. Gomi, “A computational model of four regions of the cerebellum based on feedback-error learning,” Biol. Cybern., vol. 68, no. 2, pp. 95–103, 1992.

M. Gerwig et al., “Timing of conditioned eyeblink responses is impaired in cerebellar patients,” J. Neurosci., vol. 25, no. 15, pp. 3919–3931, 2005.

D. L. Mills, “Internet time synchronization: The network time protocol,” IEEE Trans. Commun., vol. 39, no. 10, pp. 1482–1493, Oct. 1991.

A. Karmiel and G. F. Inbar, “A model for learning human reaching movements,” Biol. Cybern., vol. 77, no. 3, pp. 173–183, 1997.

N. Schweighofer, M. A. Arbib, and M. Kawato, “Role of the cerebellum in reaching movements in humans. I. Distributed inverse dynamics control,” Eur. J. Neurosci., vol. 10, no. 1, pp. 86–94, 1998.

P. van der Smagt, “Benchmarking cerebellar control,” Robot. Autonom. Syst., vol. 32, no. 4, pp. 237–251, 2000.

H. Hoffmann, G. Petkos, S. Bizer, and S. Vijayakumar, “Sensor-aided adaptive motor control under continuously varying context,” in Proc. ICINCO Autom. Robot., 2007, pp. 262–269.

S. Chitta, I. Suan, and S. Cousins, “MoveIt! [ROS topics],” IEEE Robot. Autom. Mag., vol. 19, no. 1, pp. 18–19, Mar. 2012.

S. Vijayakumar, A. D’ouza, T. Shibata, J. Conradt, and S. Schaal, “Statistical learning for humanoid robots,” Auton. Robots, vol. 12, no. 1, pp. 55–69, 2002.

C. Williams, S. Klance, S. Vijayakumar, and K. M. Chai, “Multi-task Gaussian process learning of robot inverse dynamics,” in Advances in Neural Information Processing Systems. San Diego, CA, USA: Curran Assoc., 2009, pp. 265–272.

S. Schaal, C. G. Atkeson, and S. Vijayakumar, “Scalable techniques from nonparametric statistics for real time robot learning,” Appl. Intell., vol. 17, no. 1, pp. 49–70, 2002.

E. D. Wilson et al., “Cerebellar-inspired algorithm for adaptive control of nonlinear dielectric elastomer-based artificial muscle,” J. Roy. Soc. Interface, vol. 13, no. 122, 2016, Art. no. 20160547.

P. Dean, J. Porrill, C.-F. Ekert, and H. Jörntell, “The cerebellar microcircuit as an adaptive filter: Experimental and computational evidence,” Nat. Rev. Neurosci., vol. 11, no. 1, pp. 30–43, 2010.

S. Vijayakumar and S. Schaal, “Locally weighted projection regression: An o (n) algorithm for incremental real time learning in high dimensional space,” in Proc. 17th Int. Conf. Mach. Learn. (ICML), 2000, pp. 288–293.

T. Sinkjær, C. Wu, A. G. Barto, and J. C. Houk, “Cerebellar control of endpoint position-a simulation model,” in Proc. Int. Joint Conf. Neural Netw.(IJCNN), 1990, pp. 705–710.

F. Chao, D. Zhou, C.-M. Lin, L. Yang, C. Zhou, and C. Shang, “Type-2 fuzzy hybrid controller network for robotic systems,” IEEE Trans. Cybern., to be published.
C. Nie, Z. Zheng, and M. Zhu, “Three-dimensional path-following control of a robotic airship with reinforcement learning,” Int. J. Aerosp. Eng., vol. 2019, Feb. 2019, Art. no. 7854173.

J. Cao, W. Liang, J. Zhu, and Q. Ren, “Control of a muscle-like soft actuator via a bioinspired approach,” Bioinspiration & Biomimetics, vol. 13, no. 6, 2018, Art. no. 066005.

S. Tolu, M. Vanegas, J. A. Garrido, N. R. Luque, and E. Ros, “Adaptive and predictive control of a simulated robot arm,” Int. J. Neural Syst., vol. 23, no. 3, 2013, Art. no. 1350010.

A. Lenz, S. R. Anderson, A. G. Pipe, C. Melhuish, P. Dean, and J. Porrill, “Cerebellar-inspired adaptive control of a robot eye actuated by pneumatic artificial muscles,” IEEE Trans. Syst., Man, Cybern. B, Cybern., vol. 39, no. 6, pp. 1420–1433, Dec. 2009.

H. T. Kalidindi, T. G. Thruruthel, C. Laschi, and E. Falotico, “Cerebellum-inspired approach for adaptive kinematic control of soft robots,” in Proc. 2nd IEEE Int. Conf. Soft Robot. (RoboSoft), 2019, pp. 684–689.

A. Bouganis and M. Shanahan, “Training a spiking neural network to control a 4-dof robotic arm based on spike timing-dependent plasticity,” in Proc. Int. Joint Conf. Neural Netw. (IJCNN), 2010, pp. 1–8.

A. Antonietti, D. Martina, C. Casellato, E. D’Angelo, and A. Pedrocchi, “Control of a humanoid NAO robot by an adaptive bioinspired cerebellar module in 3D motion tasks,” Comput. Intell. Neurosci., vol. 2019, Jan. 2019, Art. no. 4862157.

S. Wang, W. Chaowalitwongse, and R. Babuska, “Machine learning algorithms in bipedal robot control,” IEEE Trans. Syst., Man, Cybern. C, Appl. Rev., vol. 42, no. 5, pp. 728–743, Sep. 2012.

J. P. Hwang and E. Kim, “Robust tracking control of an electrically driven robot: adaptive fuzzy logic approach,” IEEE Trans. Fuzzy Syst., vol. 14, no. 2, pp. 232–247, Apr. 2006.

Z. Bing et al., “End to end learning of spiking neural network based on R-STDP for a lane keeping vehicle,” in Proc. IEEE Int. Conf. Robot. Autom. (ICRA), 2018, pp. 1–8.

D. D. Ligutan, A. C. Abad, and E. P. Dadios, “Adaptive robotic arm control using artificial neural network,” in Proc. IEEE 10th Int. Conf. Humanoid Nanotechnol. Inf. Technol. Commun. Control Environ. Manag. (HNICEM), 2018, pp. 1–6.

R. de Azambuja, A. Cangelosi, and S. V. Adams, “Diverse, noisy and parallel: a new spiking neural network approach for humanoid robot control,” in Proc. Int. Joint Conf. Neural Netw. (IJCNN), 2016, pp. 1134–1142.

T. Flash and N. Hogan, “The coordination of arm movements: An experimentally confirmed mathematical model,” J. Neurosci., vol. 5, no. 7, pp. 1688–1703, 1985.

Ignacio Abadía received the B.Sc. and M.Sc. degrees in telecommunications engineering and secondary education teaching from the University of Granada, Granada, Spain, in 2015 and 2016, respectively. In 2018, he joined the Applied Computational Neuroscience Research Group, University of Granada, where he was awarded a Young Researchers Fellowship. His current research interests include neuromorphic engineering, spiking neural networks, brain–computer interfaces, and motor control.

Francisco Naveros received the first M.Sc. degree in telecommunication, the second M.S. degree in computer science and networks, and the Ph.D. degree in computational neuroscience from the University of Granada, Granada, Spain, in 2011, 2012, and 2017, respectively.

Since 2017, he has been a Postdoctoral Researcher with the Applied Computational Neuroscience Research Group, University of Granada. He has authored eight articles. His current research interests include biologically processing control schemes, parallel and real-time spiking neural network, and compliant robots.

Jesus A. Garrido received the M.Sc. and Ph.D. degrees in computer science and the M.Sc. and Ph.D. degrees in computer engineering and networks from the University of Granada, Spain, in 2006, 2007, and 2011, respectively.

From 2012 to 2015, he was with the Brain and Behavioral Science Department, University of Pavia, Pavia, Italy, under supervision of Prof. D’Angelo. He has authored over 25 articles. His current research interests include cerebellar information processing and learning, motor control, neuromorphic engineering, and spiking neural networks.

Dr. Garrido was awarded a Young Researchers Fellowship by the University of Granada, in 2015. From 2016 to 2019, he obtained an IF Marie Curie Postdoctoral Fellowship from the EU with the Applied Computational Neuroscience Research Group, University of Granada.

Eduardo Ros received the M.Sc. and Ph.D. degrees in physics and computational neuroscience from the University of Granada, Granada, Spain, in 1992 and 1997, respectively.

He is currently a Full Professor with the Department of Computer Architecture and Technology, University of Granada, where he is the Head of the Applied Computational Neuroscience Research Group. He has authored over 85 scientific articles. His current research interests include bioinspired processing, neuromorphic engineering, spiking neural networks, and computational neuroscience.

Nieto R. Luque received the B.Sc. degree in electronics and the M.Sc. degree in automatics and industrial electronics from the University of Córdoba, Córdoba, Spain, in 2003 and 2006, respectively, and the M.Sc. and Ph.D. degrees in computer science and networks from the University of Granada, Granada, Spain, in 2007 and 2013, respectively.

From 2015 to 2017, he obtained an IF Marie Curie Fellowship from the EU with Dr. Arleo’s Lab, Paris, France. In 2018, he obtained a Juan de la Cierva Incorporation Post-Doctoral Fellowship from the Spanish Government with the Applied Computational Neuroscience Research Group, University of Granada. He has authored over 20 articles. His current research interests include biologically processing control, spiking neural networks, and aging.