Machine Learning Capabilities of a Simulated Cerebellum

Matthew Hausknecht, Wen-Ke Li, Michael Mauk, and Peter Stone

Abstract—This article describes the learning and control capabilities of a biologically constrained bottom-up model of the mammalian cerebellum. Results are presented from six tasks - eyelid conditioning, pendulum balancing, PID control, robot balancing, pattern recognition, and MNIST handwritten digit recognition. These tasks span several paradigms of machine learning including supervised learning, reinforcement learning, control, and pattern recognition. Results over these six domains indicate that cerebellar simulation is capable of robustly identifying static input patterns even when randomized across the sensory apparatus. This capability allows the simulated cerebellum to perform several different supervised learning and control tasks. On the other hand, reinforcement learning and temporal pattern recognition both prove problematic due to the delayed nature of error signals and the simulator's inability to solve the credit assignment problem. These results are consistent with previous findings which hypothesize that in the human brain, the basal ganglia is responsible for reinforcement learning while the cerebellum handles supervised learning.

Index Terms—Cerebellum, Inverted Pendulum Balancing (Cart-Pole), PID Control, Cerebellar Pattern Recognition, Robot Balance, MNIST Handwritten Digit Recognition

1 Introduction

Comprising only 10% of total brain volume but containing more neurons than the rest of the brain put together, the cerebellum contributes to coordination, precision, and accurate timing of movements [1], [2], [3]. The cerebellum's well characterized synaptic organization and physiology make it a good candidate for computational simulation. Additionally, tasks such as eyelid conditioning and the vestibulo-ocular reflex are known to engage the cerebellum directly and provide a source of data against which cerebellum simulations can be validated and tuned.

This article applies a biologically constrained cerebellum simulation to a variety of different types of machine learning tasks including supervised learning, reinforcement learning, and sequential pattern recognition. Conclusions are drawn about the machine learning capabilities of the cerebellar model given its performance on each different category of task.

Throughout this article, care is taken to differentiate between conclusions specific to the cerebellum simulator and the actual cerebellum. All experiments are performed in simulation, and all conclusions are directly applicable to the cerebellum simulator. In some cases, enough evidence of actual cerebellar function is present to extend conclusions to the physical cerebellum. The cerebellum simulator is a well tested but not perfect model of the physical cerebellum and conclusions about the "cerebellum simulator" apply only to the model and have not been validated on the real cerebellum.

1.1 Related Work

The cerebellum simulator used in this article is based on the Marr-Albus-Ito [4], [5], [6] theory of cerebellar function. Alternative theories of cerebellar function have been proposed such as Wolpert's theory that the cerebellum learns to replace reflexes with a predictive controller using both forward and inverse controllers [7], Kawato's theory of internal models [8], Houk's theory of the cerebellum as an adjustable pattern generator [9], [10], and Llinás' tensor geometrization theory [11], [12].

In several cases, Cerebellar simulations have been used in the context of robotic control [13]. Specific applications include cerebellar control of a robotic arm [14] and learning to time when to release an actuator to throw a ball [15].

Kettner et al. [16] first introduced the idea of using synaptic eligibility traces as a mechanism to bring the temporal gap between predictive signals and subsequent reflexive motor responses. Building on this work, McKinstry et al. [17], [18] use cerebellar models for predictive robot control tasks such as navigating a curved path. In order to achieve *predictive* control, these models employ an eligibility trace which increases eligibility for plasticity for certain synapses after a fixed delay from the onset of suprathreshold presynaptic activity. The limitation of this method is the need for a fixed temporal delay to be specified for each task. Experiments show that tasks are highly sensitive to this delay. Eligibility traces were not used in this article due to the lack of evidence that the cerebellum uses such learning mechanisms.

The idea of using a cerebellum simulator to balance an inverted pendulum was described by Ruan et al in [19], in which a cerebellar neuronal network worked in conjunction with a feedback (PD) controller to balance a two-wheeled robot. The resulting system proved stable and robust to abrupt changes to the load the robot was carrying. The cerebellar network studied by [19] consisted of only 128 cells, far fewer than in this article. Additionally, Ruan uses cerebellar network to fine-tune motor commands generated from the PD controller while this article uses the cerebellar network to initiate and control all motor movement.

Yamazaki et al [20] used a gpu-enabled cerebellum simulator containing 100,000 neurons to learn the correct time to swing a bat to hit a baseball. While the demonstration is eye catching, this

M. Hausknecht and P. Stone; Department of Computer Science, University of Texas at Austin; {mhauskn,pstone}@cs.utexas.edu.
W. Li and M. Mauk; Center for Learning and Memory, University of Texas at Austin; wenke.li@utexas.edu, mmauk@mail.clm.utexas.edu.

task itself is identical to eyelid conditioning, a known cerebellar benchmark task. A more ground breaking would be to incorporate learning of not just when to swing, but where and how hard to swing. The control problems outlined in this work all require more sophisticated control paradigms.

1.2 Main Contributions

This article contributes an investigation of the machine learning capabilities of a simulated cerebellum, characterizing its strengths and weaknesses along the dimensions of pattern recognition/supervised learning, control, and Reinforcement Learning. Of these paradigms, the simulator is strongest on supervised learning and weakest on Reinforcement Learning. To better understand this weakness, this article contributes a novel analysis of the causative factors underlying the cerebellum simulator's shortcomings on Reinforcement Learning tasks.

1.3 Organization

Section 2 describes the cerebellum simulator and previous application to eyelid conditioning (Section 3). Sections 4-9 extend the simulator to novel tasks, respectively: inverted pendulum balancing, PID control, robot balancing, static pattern recognition, temporal pattern recognition, and MNIST handwritten digit recognition. Task-specific results and analysis are presented within each section. Discussion is presented in Section 10. Section 11 examines future work and concludes.

2 Materials and Methods

Broadly, this section describes the organization of the human cerebellum and the simulator used to capture its essential computations and learning rules. The descriptions remain at a high level, but a curious reader may find the specific equations and parameters underlying the simulation in the Appendices.

2.1 Cerebellum Synaptic Organization

The cerebellum comprises a network of cells with known sites and rules for plasticity, numerical ratios, convergence/divergence ratios, and geometry of projections [21], [22], [23]. It contains an enormous number of neurons but a limited number of neuron types with a known connectivity (Figure 1).

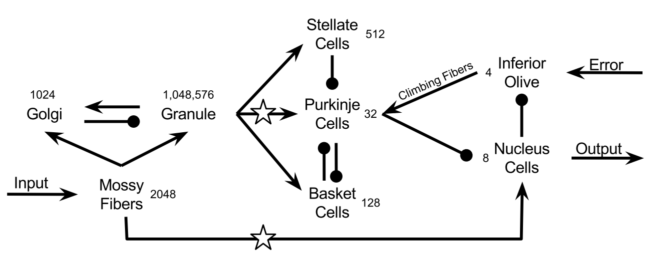


Fig. 1: Connectivity and scale of the simulated cerebellum. Arrows and circles respectively denote excitatory and inhibitory synaptic connections. The number of simulated cells in each region is annotated. Stars denote sites of synaptic plasticity.

The mossy fibers serve as a bridge for information to flow into the cerebellum and carry information about the state of the world. Similarly, error or teaching signals originate in the Inferior Olive and are transmitted via Climbing Fibers. These errors signal the need for changes in synaptic plasticity and ultimately changes in behavior. Behavioral changes are manifested in nucleus cell outputs which form the basis of muscle control. Comprising half the total neurons in the human brain [3], granule cells also play a key role cerebellar learning.

The cerebellum learns by updating synaptic strengths of neurons according to two known pathways: In the first pathway, mossy fibers increase output responses via direct excitatory connections onto the deep nuclei. In the second pathway, the granule to Purkinje excitatory synapses are modified according to the climbing fiber inputs such that the synapses active shortly prior to the climbing fiber input decrease weight, which causes the Purkinje cells to reduce their activity the next time the same input is encountered. The decrease in Purkinje cell activity then induces the mossy fiber to nucleus excitatory synapses to increase in weight, thereby causing the nucleus cells to become more responsive to the same mossy fiber inputs [24]. Synaptic modification at these two sites (denoted by stars in Figure 1) forms the basis of *feed forward prediction*, believed to underlie the cerebellum's ability to coordinate and fine-tune motor responses [25].

The cerebellum may also be understood as an artificial neural network which receives a vector of input and a scalar error signal at each timestep and produces an output vector of nucleus cell firings. It is better described as a recurrent neural network due to the directed cycles of between the granule-golgi, Purkinje-basket, and olive-Purkinje-nucleus cells. These cycles feature inhibitory as well as excitatory connections and allow the network to exhibit dynamic temporal behavior. The learned parameters of the network are the synaptic weights between granule and Purkinje cells as well as the weights between mossy fibers and nucleus cells. These parameters are updated every timestep through a process similar to Hebbian learning. The next section describes the process of computationally simulating the activity of the cerebellum.

2.2 Cerebellum Simulator

Computer modeling of the cerebellum has been a subject of active research for over a decade, with increasingly detailed models continually being created [26], [27], [28], [17]. This article uses a biologically-constrained, bottom-up cerebellar model based on the model introduced by Buonomano and Mauk [29] and by Medina and Mauk [27] based on the Marr-Albus-Ito [4], [5], [6] theory of cerebellar function. Compared to [27], the cerebellum simulation used in this article has nearly two order of magnitude more granule cells, from 12,000 to 1,048,576. Consequently, the divergence/convergence ratios more closely approximate those observed in the real cerebellum. The sheer number of cells in the human cerebellum still dwarfs the simulation by more than four orders of magnitude.

The simulator uses Nvidia graphics processing units (GPUs) to parallelize computation of granule cell firings. Speedups from traditional parallel programming approaches such as OpenMP were inadequate due to the high memory bandwidth required to compute firings, roughly 128 GB/s for the simulation to run at real time speed. However, since calculating granule cell activities involves applying identical equations to each cell, vector processors like GPUs are particularly well-suited for such tasks.

Compared to other cerebellum models, this model has three distinct advantages: 1) Instead of modeling high-level cerebellar learning [15], each cellular region is directly modeled after the observed physiology and connectivity of the actual cerebellum. 2)

The simulator has been validated against and shown to replicate animal behavior data collected on Eyelid Conditioning. 3) This simulator is an order of magnitude larger than prior models (neurons in prior models: [13]:1, [19]:6, [18]:1101, [14]:1500, [16]:6000, [20]:100,000) Experiments show that larger models increase fit to observed animal data for certain CS-US intervals [30], indicating that increasing model complexity pays off in increased representational power.

All experiments in this article use this cerebellum simulator with the same number of cells (1,051,308), the same connectivity (Figure 1), and the same learning rules. The equations and updates required to simulated learning are given in the Appendices.

3 Background: Eyelid Conditioning

The quality of a neuroscience model is contingent upon its ability to recreate and ultimately predict experimental data. This section describes eyelid conditioning, the original task against which this simulator was tuned and validated. Eyelid conditioning is not a contribution of this paper but illustrates principles of cerebellar learning which will be leveraged throughout the article.

Eyelid conditioning is a form of classical conditioning known to have direct ties with the cerebellum [31]. The procedure involves pairing a sensory stimulus (the conditioned stimulus (CS)) with an eyeblink-eliciting unconditioned stimulus (US), typically an air puff directed at the eye or peri-orbital electrical stimulation. Untrained animals initially produce a reflexive, unconditioned response (UR) (e.g. blink or extension of nictitating membrane) that follows US onset. After many CS-US pairings, an association is formed such that a learned eyelid response, or conditioned response (CR), occurs and precedes US onset. These conditioned responses are not permanent and in a phenomenon known as *extinction*, learned condition responses subsequently subside if a conditioned animal is presented with the CS unpaired with the US. When paired presentations are reintroduced following extinction, CRs reappear far more quickly than they were initially learned, a phenomenon known as *savings*.

Eyelid conditioning experiments were used to evaluate and tune the cerebellum simulation by providing CS-like and US-like inputs over mossy fibers and climbing fibers respectively while ensuring that the simulator's learning of conditioned responses reflected data collected from test subjects. Thus, the rich repertoire of well-characterized behavioral properties of eyelid conditioning were leveraged as a stringent test of the simulation accuracy.

For example, the CS-US interval, or the amount of delay between the incidence of the CS and the following US, greatly influences a rabbits ability to learn conditioned responses. CS-US intervals shorter than 100ms as well as intervals longer than 1500ms were found to preclude learning. Previously, Medina and Mauk described a cerebellum simulator capable of learning conditioned responses for a variety of CS-US intervals [27]. In most cases the cerebellar simulation described in this article mirrors their results. However, for the 1000ms CS-US interval case, consistent with rabbit data, the current simulation, unlike the previous one, successfully learns conditioned responses [30].

Ultimately, the quality of a simulation depends not only on its ability to recreate experimental data upon which it has been tuned, but on its ability to handle novel tasks and make biologically relevant predictions. A main contribution of this article is expanding the validation of the simulator on new control tasks. In the next section, the simulator is applied, without further tuning of internal parameters, to the inverted pole balancing task.

4 Pole Balancing

Pole-balancing experiments extend previous work [30] and present novel analysis and results. Inverted pendulum balancing is a well-known control benchmark [32], [33] that involves a pole affixed to a cart via a joint, forming an inherently unstable system in which the pole mass is located above the pivot point. Active balance of the pendulum is achieved by applying force to the cart – resulting in horizontal movement of the pivot point and angular rotation of the pole. The objective is to balance the pendulum for as long as possible. Unlike eyelid conditioning which requires only a single force (closure of the eyelid) in response to a single input (auditory tone), inverted pendulum balancing requires the coordination of multiple forces in response to multiple sensory inputs. Despite the added complexity, inverted pendulum balancing is particularly suited for cerebellar learning because it involves reactive balance and predictive control.

4.1 Cerebellum-Cartpole Interface

Though the cerebellum simulator models in detail what goes on *inside* the cerebellum, it does not specify the interfaces between the cerebellum and the environment and indeed the nature of such an interface is unknown. This section describes the mechanisms interfacing the cerebellum simulation to the external world. Specific focus is placed on conveying state and error information to the cerebellum simulator and interpreting the cerebellar firing rates as actions applicable to the inverted pendulum domain. The overall architecture of this interface is depicted in Figure 2.

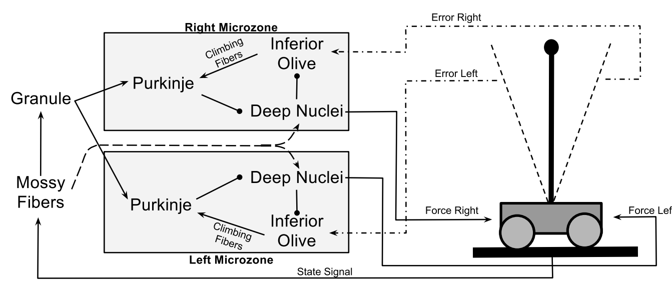


Fig. 2: Cerebellum-Cartpole Interface: the environment transmits state and error signals to the mossy fibers and inferior olivary nuclei. In return, the cerebellum simulator provides real-valued output from two microzones which is applied to the cart as force in opposite directions. Arrows and circles respectively denote excitatory and inhibitory connections.

In each simulated timestep, the cerebellum simulator receives a description of the current state of the system, performs an action, and receives information about the resulting state and reward. This cycle repeats until the trial has terminated – either by the pole falling or remaining balanced for 1 million simulated timesteps (approximately 16 minutes of real time). Architectural modifications known as *microzones* were required to achieve multiple output forces necessary for inverted pendulum balancing. Anatomically, the cerebellar cortex is believed to be divided into functionally distinct regions called *microzones*, each of which controls a specific muscle group [34]. Because of the regular patterns of connectivity within each microzone, it is thought that different microzones exhibit similar learning mechanisms and differ only in their control of different muscle groups. As Figure 2 indicates, each microzone contains a full set of Purkinje, nucleus, and inferior olive cells, but shares common input cells such as mossy fibers and granule cells. Multiple microzones are essential for control tasks with more than one degree of freedom. In pole balancing the simulator uses two microzones to push the cart in each of the two possible directions along the track.

State Encoding: As input, mossy fibers process state signals received from the cartpole domain. State signals were chosen to include the angle θ and angular velocity $\dot{\theta}$ of the pole as well as the location x and velocity $\dot{x}$ of the cart on the track. Of the 1024 mossy fibers (MFs) present in the cerebellum simulation, 30 random non-contiguous mossy fibers were allocated to encode each of the four state variables. A Gaussian distribution was created for each of the 30 MFs with means μ distributed evenly over the range of values associated with the corresponding state variable. Given the current value of the state variable x, the boolean firing of each MF_i is sampled from $\frac{1}{\sigma_i\sqrt{2\pi}}e^{-\frac{1}{2}(\frac{x-\mu_i}{\sigma_i})^2}$ where μ_i and $\sigma_i = \sqrt{.2 \cdot \text{range}(x)}$ are the mean and standard deviation of MF_i's normal distribution.

Error Encoding: Error signals enter the cerebellum simulation through the inferior olivary cells located in each of the two microzones. Typically in the inverted pendulum domain negative reinforcement is delivered to the agent only when the pole falls or the cart leaves the track [32]. However, due to issues with extinction, error was probabilistically delivered at each timestep prior to the pole falling or the cart leaving the track. The following equations specify the probability of the right-pushing microzone (Fig 2) receiving an error on a given timestep.[1] These equations are symmetric for the left microzone.

$$p(Err_\theta) = \begin{cases} 0 & \text{if } \theta < 0 \text{ and } \dot{\theta} < 0 \\ \min(\text{abs}(\theta), .01) & \text{otherwise} \end{cases} \tag{1}$$

$$p(Err_x) = \begin{cases} 0 & \text{if } \theta < 0 \text{ and } x < 0 \\ \min(\text{abs}(x), .01) & \text{otherwise} \end{cases} \tag{2}$$

$$p(Err_{\dot{x}}) = \begin{cases} 0 & \text{if } \dot{x} < 0 \\ \min(\text{abs}(\dot{x}), .01) & \text{otherwise} \end{cases} \tag{3}$$

Equations 1-3 are sampled independently and an error is delivered to the microzone if any of the three triggers. However, Microzones detect only the presence or absence of error and cannot distinguish between the different underlying causes of the error: $Err_\theta, Err_x, Err_{\dot{x}}$ (respectively pole angle, cart position, cart velocity errors). Thus error signals are ambiguous in nature.

Output Encoding: As output, the eight deep nuclei of each microzone encode forces applied to the cart in opposite directions. Each force is extracted as the average firing rate of the 8 deep nuclei, yielding a [0,1] continuous value. The force is then scaled and applied to the cart.

4.2 Pole Balancing Experiments and Results

Cerebellar pole balancing performance was compared against two other agents: a naïve agent and a Q-Learning agent. The naïve agent applies a force to the right when the pole angle is larger than zero and a force to the left when the pole angle is less than zero. This controller is sufficient to keep the pole balanced but eventually fails due to the limited track. Next, Q-Learning was chosen as a comparison point because of its popularity and simplicity, despite the fact that it is by no means a state of the art reinforcement learning algorithm. The Q-Learning agent used a state encoding as similar as possible to that of the cerebellum simulator. Pole angle and velocity as well as cart position and velocity were each encoded using a 10-tiled CMAC tile coding scheme, originally proposed by Albus and motivated by the cerebellum [35]. Additionally, error was delivered to Q-Learning agent in the same manner following Equations 1-3. Each of the parameters was experimentally tuned to maximize Q-Learning performance, however the Q-Learning agent generally required nearly a thousand trials before it was able to balance the pole for a million cycles. Figure 3 compares the pole balancing performance of cerebellar, naïve, and Q-Learning agents. Surprisingly, the cerebellum is able to learn highly successful policies within the first five trials. After as few as eight trials, perfect performance is achieved: the cart can both balance the pole and remain centered on the track. As the next section discusses, this success results in part from the regular error signals that are delivered while the pole is falling or the cart is nearing an edge of the track.

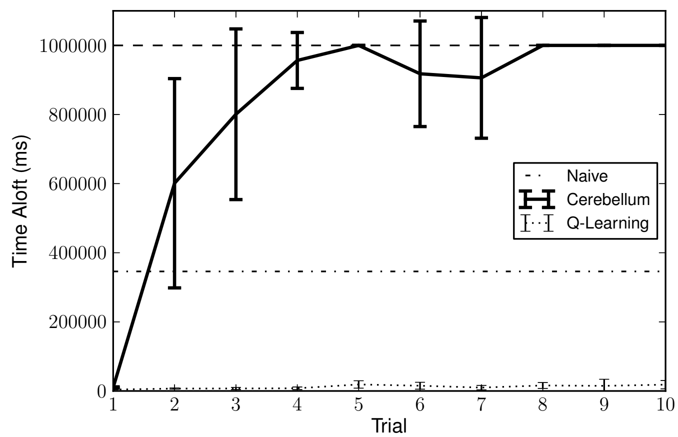


Fig. 3: The cerebellum simulator solves the inverted pendulum task within eight episodes. Q-Learning eventually converged to the correct solution after nearly a thousand episodes.

4.3 Necessity of Regular Error Signals

Unlearning occurs in the prolonged absence of regular error signals and is characterized by diminished responses to previously trained stimuli. There are two principle causes of unlearning: *forgetting* and *extinction*. Forgetting is caused by the accumulated synaptic weight drift that occurs continually when an organism is not learning the task. Extinction is a deliberate type of unlearning that occurs when a conditioned stimulus is presented but not paired with an unconditioned stimulus (error signal). Spontaneous climbing fiber activity allows the cerebellum to retain learned responses in the absence of error signals. Extinction works by suppressing this spontaneous climbing fiber activity.

Experiments throughout this article indicate that in order to avoid unlearning, error signals need to be delivered regularly throughout the course of a task. An example of this phenomenon was observed when error was delivered only at the end of a trial. In such a scenario, the cerebellum learns to balance the pole after receiving several errors. Good balance is retained for around 15,000 timesteps but slowly, the learned responses diminish in

1. θ and $\dot{\theta}$ correspond to the pole angle and velocity (radians) while x and $\dot{x}$ are the cart position and velocity. $\{\theta, \dot{\theta}, x, \dot{x}\} = 0$ corresponds to a cart centered on the track with pole upright. Negative values indicate the pole falling left or the cart on the left side of the track.

force, until the pole again falls. This cycle of learning and falling followed by balance and unlearning continues indefinitely. Unlearning is a fundamental aspect of eyelid conditioning and cerebellar computation. Equation 9 allows unlearning through the synaptic plasticity step size δ_{-}^{gr} and δ_{+}^{gr}. While there are currently no methods for performing inverted pendulum balancing experiments on animals, the simulated results strongly predict that such experiments would require regular errors signals.

5 PID Control

As a general hypothesis, the cerebellum should be capable of performing supervised control tasks featuring regular error signals. The pole-balancing domain can be thought of as a specific instance of a setpoint control task in which the desired setpoint is a vertical pole angle. Setpoint control tasks are common in industrial and robotic settings and are typically solved by proportional-integral-derivative controllers (PID controllers) [36]. One example PID control task is controlling the acceleration and deceleration of an autonomous vehicle in order to reach a desired velocity. This task is accomplished with two PID controllers - one controlling the brakes and the other controlling the gas pedal.

The cerebellum was adapted to this task using the equations of motion derived from a simulation of Austin Robot Technology's Autonomous vehicle [37]. The equations capture factors such as rolling resistance and wind resistance.[2]

The task was formulated as an episodic Markov Decision process in which each trial starts with a randomly generated current and target velocity in the range of $(0 - 11)$ meters-per-second. Each trial lasts for 10-seconds of simulated time. The agent's reward at each simulated time step is -10 times the absolute value of the difference between the target and current velocity of the car.

The cerebellum simulator received a state signal indicating the difference between the current and target velocity and had to choose to either apply force to the accelerator or the brake. Two Microzones are used to actuate these controls. Simultaneous activation of the accelerator and brake was not allowed and if both Microzones had non-zero output force, the Microzone with the higher output force would activate the corresponding pedal with a force equal to the magnitude of the difference between the two forces. An error signal was given to either the acceleration or brake Microzone with a probability proportional to the difference between the current and target velocity.

5.1 PID Control Results

The performance of the cerebellum simulator was compared against a model-based Reinforcement Learning algorithm called TEXPLORE [38], [39] as well as two different PD controllers: *Tuned PD*, a tuned controller which represents an upper bound for performance, and *Online PD*, a controller in which parameters were optimized using hill climbing.

Figure 4 shows the resulting performance. The cerebellum simulator learns quickly and shows the best performance in the first 400 episodes, but is eventually overtaken by the Online PD controller. TEXPLORE's performance improves, but is not competitive within the provided training time.

2. The code for the domain can be viewed at https://code.google.com/p/rl-texplore-ros-pkg/source/browse/trunk/stacks/reinforcement_learning/rl_env/src/Env/RobotCarVel.cc.

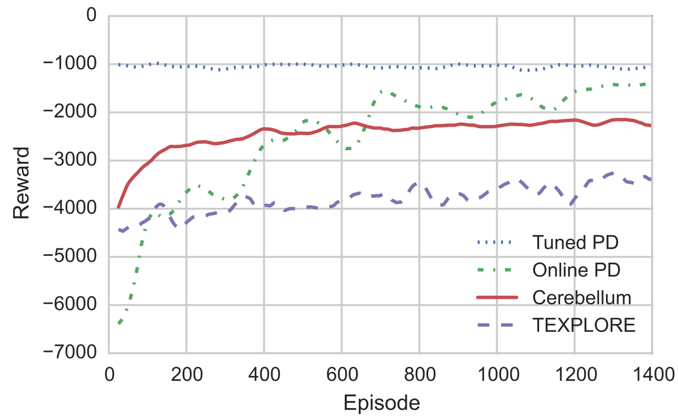


Fig. 4: Simulated Autonomous Vehicle Control: Average reward, in order of final performance, of a pre-tuned PD controller, online PD controller, the cerebellum simulator, and TEXPLORE, on the simulated autonomous vehicle acceleration and deceleration control task. PD controller was pre-tuned automatically using hill climbing. Results are averaged over ten trials and smoothed over a 50 episode sliding window.

The performance difference between the cerebellum simulator and the PD controller is largely due to the higher gains achieved by the PD controller. Both the cerebellum simulator and the PD controller eventually get close to the desired velocity, however the PD controller is capable of doing so much faster and with greater precision than the cerebellum simulator.

To illustrate this difference, a sample 2-target-point task was created in which the vehicle was asked to accelerate to a velocity of ten meters per second and after ten seconds, to decelerate back to the original velocity of five meters per second. Figure 5 shows that both controllers approximately complete the task, however the PD controller is much more precise in its application of the accelerator and the brakes. These differences account for the performance gap shown in Figure 4.

One possible advantage of using cerebellar control is not having to manually tune parameters for a PID controller, which could be valuable if the dynamics of the task at hand are unknown. These experiments give credence to the idea that the cerebellum is capable of performing PID related tasks to some degree of precision. Broadly, control tasks featuring regular, supervised error signals such as inverted pendulum balancing and acceleration control are well-suited for simulated cerebellar learning.

6 Dynamic Robot Balance

In all of the domains discussed previously, error signals were delivered at the exact point in time that more output force was necessary. In pole balancing, error signals were delivered as the pendulum was falling; in PID control, error signals were delivered with frequency proportional to the difference between the current and the target point. From a learning perspective these signals correspond to supervisory signals as they tell the cerebellum simulator if it needs to output more force at that moment (and any future time in which similar state inputs are active).

A more complex form of error signal is found in reinforcement

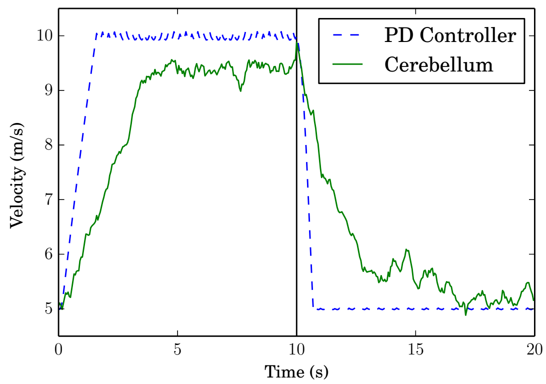


Fig. 5: Sample performance of acceleration and braking: A hill-climbing tuned PD controller is compared against the cerebellum simulator in a task designed to test the acceleration and braking capabilities of each algorithm. At time zero, a target velocity of $10m/s$ is given with a current velocity of $5m/s$. At 10 seconds, the target velocity $5m/s$ is given. The cerebellum simulator gently approaches the target velocity yet still has oscillations. The PD controller applies the maximum allowed braking and acceleration to quickly reach the target speed. Small oscillations at the target velocity can be seen. These oscillations are likely caused by the lack of an integral term.

learning in which errors[3] may occur after the point at which more output force is required. This delay gives rise to the credit assignment problem [40] in which a learning algorithm must propagate the influence of delayed reinforcement back to the states and actions which were responsible for that reinforcement.

This section presents a simulated dynamic robot balancing task which uses delayed error signals and necessitates predictive control. Dynamic robot balancing is an important and largely unsolved challenge universally encountered by bi-pedal robots. Since the human cerebellum is known to be involved in fine motor tasks such as maintaining balance while walking [41], it is reasonable to hypothesize that the cerebellum simulator should be capable of performing this task. To foreshadow our results, it was found that the cerebellum simulator has the ability to predict delayed error signals but cannot correctly time its force output to prevent them. These results support prior hypotheses that the cerebellum specializes in supervised learning rather than Reinforcement Learning [42].

6.1 Dynamic Balance Setup

The objective of this task is to maintain robot balance after the application of sudden force. Simulated robots are modeled in SimSpark[4] after the humanoid Aldebaran Nao. Sudden force is created by shooting a weighted soccer ball at the front of the robot. In order to maintain balance, a single Microzone controls the robot's forward hip pitch - meaning that high cerebellar response causes the robot to lean forward. Since impact of the ball always comes from the front, the simulator must lean forward just before the collision of with the ball, then lower its response to return the hips to a neutral position. If the robot does not return to neutral hip angles quickly, the rocking motion resulting from the impact of the ball causes a forward fall.

The task proceeds in phases: first, the robot is given two seconds to prepare for the shot. After impact, the robot needs to remain upright for three seconds to be considered stable. At the end of the three seconds, the robot and ball are reset to their original positions. Figure 6 shows the preparation and shot phases.

As input, the cerebellum simulator receives a timer which counts down until the ball is fired. The timer is encoded using contiguous mossy fiber input and allows the cerebellum simulator to exactly predict the moment of impact. It is the only state input necessary to learn this task.

Three error encodings were explored: **Gyro Error** is proportional to the magnitude of robot's internal gyroscope's deviation from upright. **Accelerometer Error** is proportional to the magnitude of the robot's acceleration in the backwards direction. **Manual Error** is an undelayed, supervised error signal based on the difference between the current hip angle and the hip angle of a known solution. Gyro and Accelerometer error are delayed error signals because it takes 250 milliseconds before they first detect the impact of the ball. Figure 7a plots the probability of error from each encoding as a function of time to impact.

It should be noted that this task is nearly identical to Eyelid Conditioning (Section 3) except for one major factor: When using the Gyro and Acceleromter encodings, cerebellar response must precede the incidence of error by one second. For reference, in Eyelid Conditioning, responses co-occur with errors.

6.2 Dynamic Balance Results

The performance of the cerebellum simulator at the dynamic balancing task is shown in Table 1 and in video at http://youtu.be/jClYGFzUntM. The manual error encoding yielded the best cerebellar policy which learned to resist the force of impact 68% of the time. The delayed error encodings were unable to learn this task. The next sub-section analyzes why the delayed error encodings proved incapable of maintaining dynamic balance.

Error Encoding	Manual	Gyro	Accelerometer
Percentage No Fall	68%	0%	0%
Percentage Fallen Backwards	20.4%	94%	99.2%
Percentage Fallen Forwards	11.6%	6%	.8%

TABLE 1: Robocup Dynamic Balance Results: Percentage of outcomes for different error encodings over 250 trials. Each trial could end with the robot staying balanced, the robot falling forwards, or the robot falling backwards. The supervised error encoding (Manual) far outperformed the delayed error encodings.

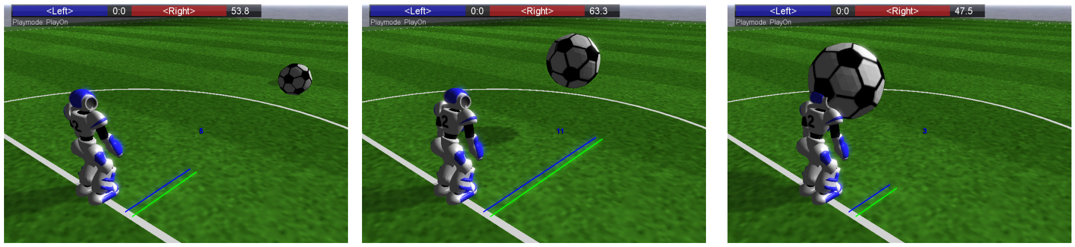
6.3 Granule Cell Analysis of Robot Balancing

The most prominent site of synaptic plasticity in the cerebellum is the Granule to Purkinje cell synapses. By tracing the learned weights of these synapses back to their Mossy Fiber inputs, it is possible to infer the relationship between a given Mossy Fiber's activation and the resulting cerebellar output force. This section introduces the *Granule Weight Measure* (GWM), a metric for predicting cerebellar response to a stimuli. GWM offers insights into the inability of simulation to maintain dynamic balance.

Each simulated Mossy Fiber connects to roughly two thousand different granule cells which in turn synapse onto 32 Purkinje cells. Mossy fibers connected to granule cells with large granule

3. In general reinforcement learning features both positive and negative feedback. In contrast, the cerebellum takes only a single type of feedback.

4. http://simspark.sourceforge.net/



(a) Preparation (b) Shot (c) Impact

Fig. 6: Dynamic Robot Balance: The objective of this task is to maintain stability in the face of sudden impact force. The cerebellum simulator controls the hip pitch of the robot, leaning forwards in preparation for impact and returning to upright after impact. A failure to do either of these results in a fall. An augmented soccer ball with increased size and mass is used to impart sudden force.

→ Purkinje weights excite connected Purkinje Cells, which inhibit nucleus cells, thus inhibiting cerebellar output. Conversely, Mossy Fibers connected to granule cells with low granule → Purkinje weights cause little Purkinje activity, little inhibition of Nucleus Cell activity, and ultimately larger cerebellar output forces. By tracing granule → Purkinje weights back to their Mossy Fiber inputs, it is possible to approximately predict the cerebellar output force resulting from the activation of each Mossy Fiber.

Specifically the *Granule Weight Measure* is defined for each Mossy Fiber to be the sum of connected granule → Purkinje weights minus the expected sum of connected granule → Purkinje weights. A positive GWM for a given Mossy Fiber indicates that any state input which triggers that Mossy Fiber will exhibit an inhibitory effect on cerebellar output force. Conversely, a negative GWM shows that a given Mossy Fiber will increase cerebellar output whenever active. Thus, it is possible to predict the cerebellum simulator's force output by examining the GWM over the Mossy Fibers spanning a state feature.

Figure 7b depicts the GWM of the *time to impact* state variable for different Robocup Error encodings. In the dynamic balance task, 307 Mossy Fibers convey information about the number of seconds until the impact of the ball (negative values represent seconds after impact). Each of the 307 Mossy Fibers is maximally activated at the corresponding value between -1.5 and 1.5 seconds to impact. The GWM for that Mossy Fiber indicates how much force the cerebellum simulator will output as a function of seconds to impact. For example, using the Manual error encoding, the GWM remains low 500 milliseconds before impact until 250 milliseconds after impact, meaning the cerebellar force output will peak and the agent will lean forward during this interval.

As can be seen in Figure 7, high error probabilities are highly correlated with low Granule Weight Measure values. However, the Granule Weight Measure values are also shifted to the left by 100-200 milliseconds. Thus whenever error is frequently delivered, the cerebellum simulator learns to output high forces for the states immediately preceding the error. This is a consequence of the plasticity update in Equation 9. In this sense, the cerebellum simulator has learned to predict the incidence of temporally delayed error signals.

However, in order to succeed at this task, the cerebellum simulator needs not only to predict the delayed error signals but also to output high force at states preceding the error. In other words, force response must be shifted to the earlier states responsible for the fall. As Figure 8 shows, the cerebellum simulator can anticipate future error signals but lacks flexibility in the timing of its responses in order to prevent them. The distinction between *supervised learning* and *reinforcement learning* parallels this point. Reinforcement learning tasks feature the *credit assignment problem* in which the agent must learn to identify the past states and actions responsible for delayed rewards. In this task at least, the cerebellum fails to respond to the states responsible for eventual error signals. Instead the cerebellum responds to the same states in which error signals occur. This response is the correct behavior in a supervised learning setting in which error signals denote labels, but not in a Reinforcement Learning setting. In future work, it may be possible to use eligibility traces [17], [18] to shift force responses to earlier states.

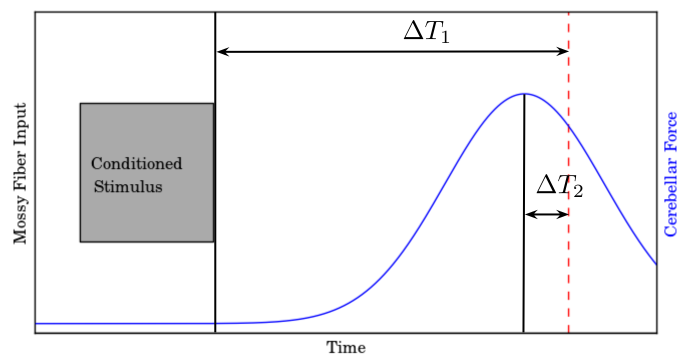
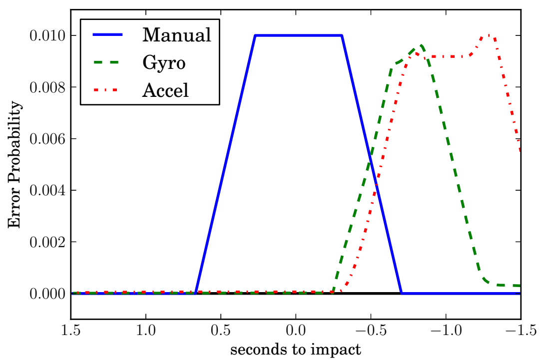


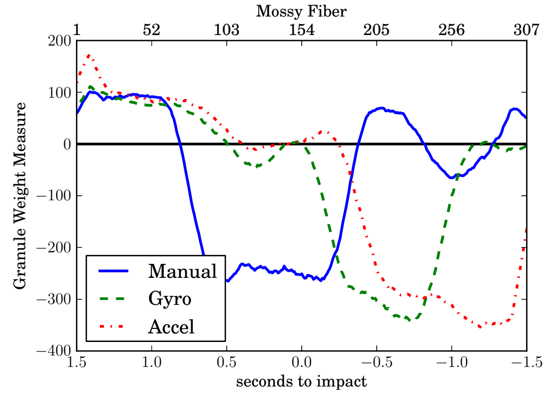
Fig. 8: The cerebellum simulator has been shown to learn robust responses over a wide variety of delays ΔT_1 ranging from 250 milliseconds to 1.5 seconds, but is inflexible in changing the delay ΔT_2 between force output and error signal. This learning window makes the cerebellum simulator suitable for control tasks featuring error signals that co-occur with high output forces. Tasks which require force output more than 100 milliseconds prior to error signal have proven difficult or impossible to learn, precluding most typical *reinforcement learning* tasks.

7 Static Pattern Recognition

Unlike most supervised learners, the cerebellum is inherently temporal by nature, meaning that rather than treating each example as an independent problem, the cerebellum takes as input a sequence of states (conveyed via Mossy Fiber activations) and has an internal state that reflects more than just the latest state input. As such it is best considered a *Sequential Supervised Learner*. Other Sequential Supervised Learners include Hidden Markov



(a) Error Probabilities



(b) Granule Weight Measure

Fig. 7: Granule Weight Analysis Shows the Effects of Delayed Error Signals: (a) Error probability as a function of time to impact. The manual encoding delivers errors at and before impact while the gyro and accelerometer encodings deliver delayed errors, after the impact is perceived. (b) The Granule Weight Measure (GWM) for the different Robocup error encodings. A lower/higher GWM corresponds to increased/decreased cerebellar output force when the associated Mossy Fiber is active. For example, in all encodings, Mossy Fiber 205 is active 0.5 seconds after impact (x=-0.5). The Gyro and Accel encodings have a highly negative GWM at this point, meaning that the robot will lean forward strongly. The Manual encoding has a positive GWM at this point meaning that the robot will lean backwards. To succeed at this task the cerebellum simulator must output high force at and directly before impact, and low force at all other times. Of the four error encodings, only the Manual encoding is capable of maintaining dynamic balance. All others output force too late or without sufficient strength. This failure is due to the delayed nature of Gyroscope, and Accelerometer error signals. Lines were smoothed using a sliding window of size ten.

Models [43] and Conditional Random Fields [44]. This section focuses on identifying the types of sequences and patterns the cerebellum simulator is and is not capable of recognizing.

In general, the cerebellum's ability to recognize a pattern of state input is largely governed by the granule cells' ability to modulate their firings as a function of temporal Mossy Fiber inputs. The following experimental setup is used to test recognition of a given function: the target function to be recognized is first encoded as Mossy Fiber input and presented for 500 milliseconds. Next a rest of 500 milliseconds is given, during which Mossy Fibers input returns to baseline. Finally, a false pattern of input is presented for 500 milliseconds followed by a rest. During training time, the target pattern is immediately followed by an error signal; the false pattern is not. During testing time, no error signals are presented for either the target or false pattern and the cerebellar output force is recorded. If the simulator has learned to recognized a function, it will present high output forces only when observing the target function and low output force otherwise.

Figure 9 shows that the cerebellum simulator is capable of recognizing all boolean functions over two input variables.[5] In each graph, the dark gray regions indicate high Mossy Fiber activation as a function of time. Error signals, presented during training time, are denoted by vertical red dashed lines that follow the target pattern. Additionally, the average cerebellar output force at test time is plotted in blue.

8 Temporal Pattern Recognition

Temporal pattern recognition is the challenge of identifying and responding to sequences of Mossy Fiber activations. In the previous section, the cerebellum simulator's decision to output high force needed to be based only on the Mossy Fiber activations of the current timestep. In contrast, in this section the cerebellum simulator must learn to respond to the Mossy Fiber activations in past timesteps as well as the current timestep. In machine learning terms, the previous section tested pure *supervised learning* capabilities while this section tests *sequential supervised learning*.

The experimental setup mirrors the previous section: at training time, the target pattern is presented followed by an error signal and a rest. Next a false pattern is presented without error signal. At test time, no error signals are presented and cerebellar output force is recorded. Success is measured by the ability to respond only to the target pattern and not to the false patterns.

Previously, Kalmbach et al. [45] demonstrated that the cerebellum is capable of learning temporal subtraction, a specific type of temporal pattern. Figure 10 shows that the cerebellum simulator used in this article successfully replicates the temporal subtraction experiment. Beyond replicating previous work, two additional temporal pattern recognition experiments are discussed in this section.

The first experiment tests the ability of the cerebellum simulator to recognize the boolean function XOR (*exclusive-or* 9) when no simultaneous Mossy Fiber activations are delivered in the same timestep. (E.g. if Mossy Fiber groups A and B are currently active, they are delivered in alternating timesteps - A on even timesteps and B on odd.) Figure 11 shows that the cerebellum simulator is still capable of learning this modified XOR function. This result indicates that granule cells can respond to Mossy Fiber inputs present on different timesteps. While this success demonstrates that the cerebellum simulator can learn from more than just the current timestep, it does not test longer-range temporal dependencies.

The second temporal experiment tests the cerebellum simulator's ability to discern between two long-ranged temporal sequences of Mossy Fiber input. Figure 12 shows that the cerebellum

5. Symmetric functions are omitted (e.g. if the cerebellum simulator can recognize stimulus $(A \wedge !B)$ it will also recognize $(B \wedge !A)$.)

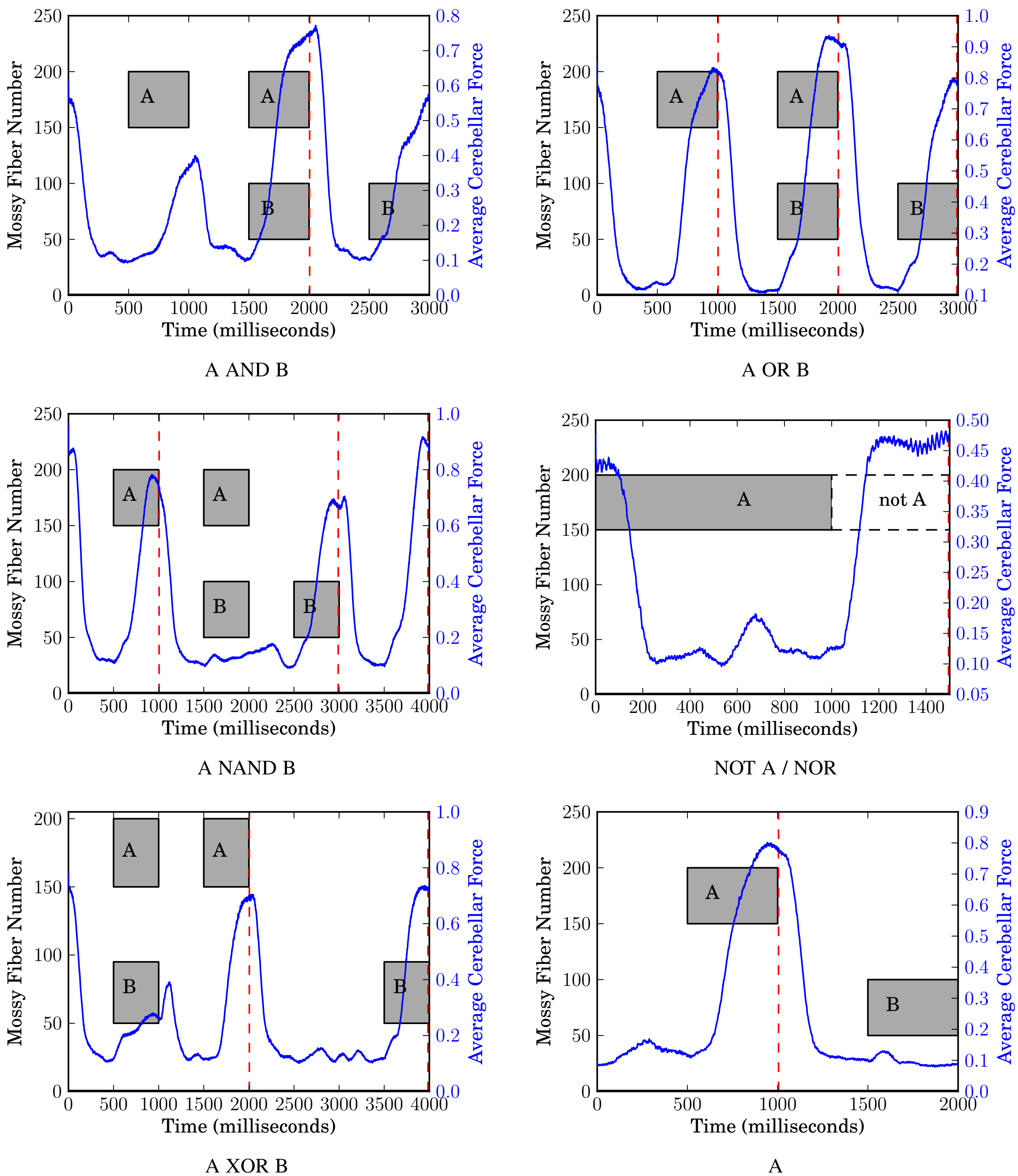


Fig. 9: Successful cerebellar learning of all two-variable Boolean functions. Blocks of contiguous high-frequency Mossy Fiber firing are shown in gray. The left y-axis of each plot shows which Mossy Fibers are active in each block. The right y-axis shows the average cerebellar output force in blue (forces were averaged over ten trials). Error signals are denoted by vertical red dashed lines. Learning is considered successful in each case because highest cerebellar output forces correspond with the incidence of error signals.

simulator has trouble identifying the difference between these two temporal patterns: high force output is manifested in response to both patterns even though training delivered error signals only following the first of the two patterns. This result suggests that the cerebellum simulator may recognize inputs cumulatively through time but not necessarily discriminatively.

From these results, the cerebellum simulator shows clear ability to learn functions over inputs spanning more than a single timestep. However, discriminating between patterns that differ only in the order in which they are presented, proves more difficult.

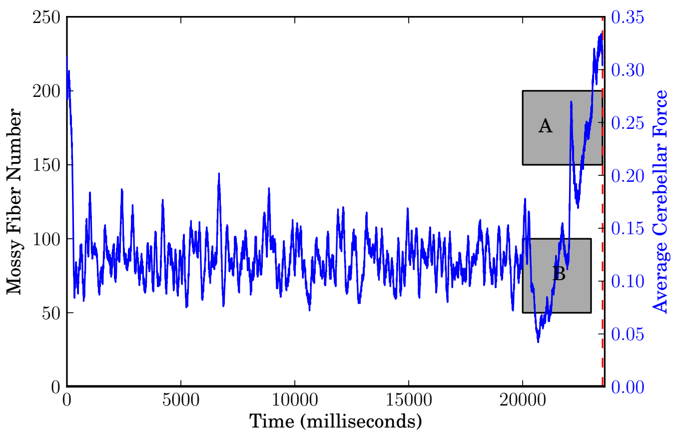


Fig. 10: The cerebellum simulator is capable of recreating the subtraction experiment described by Kalmback et al. [45]. In this experiment two overlapping tones are played. The shorter tone, here "B" is played for 3000 milliseconds while the longer one "A" is played for 3500 milliseconds. Error is delivered after the end of the longer tone. Output forces show that the simulator outputs highest force just before the error signal.

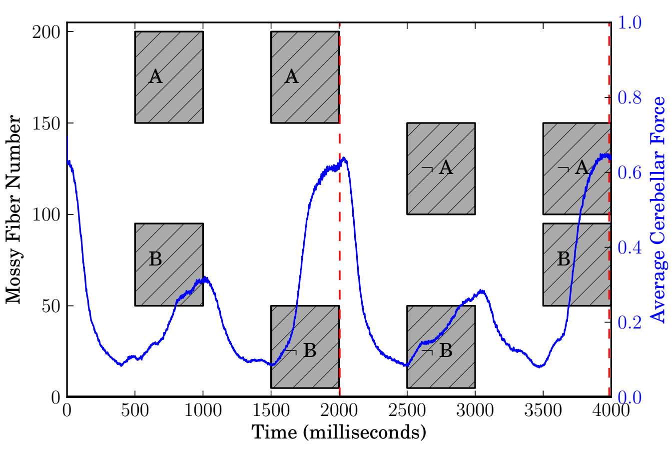


Fig. 11: The cerebellum simulator learns the *exclusive or* function when inputs are not delivered simultaneously. Dashed gray boxes indicate that Mossy Fiber inputs were delivered in alternating timesteps (e.g. A on even timesteps $\neg B$ on odd). This result indicates that granule cells can learn functions over maximally interspersed, non-overlapping sequences of Mossy Fiber input.

9 MNIST Handwritten Digit Recognition

Since the simulated cerebellum can recognize all two-variable boolean functions (Section 7), it is natural to attempt a more

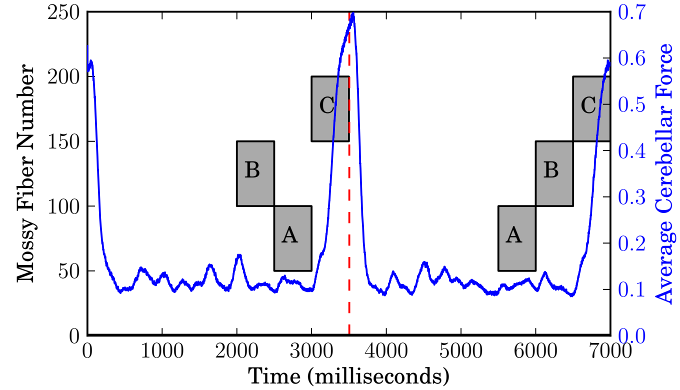


Fig. 12: Discrimination between different temporal sequences of state input: Two different sequences of Mossy Fiber activation, B, A, C and A, B, C were presented during both training and testing; only the former was followed by an error signal. Cerebellar output shows high force response to both sequences of Mossy Fiber input with slightly higher response to the correct sequence, indicating that for at least these patterns of input, the cerebellum simulator was not able to strongly distinguish the first from the second.

challenging static pattern recognition task featuring supervised errors such as handwritten digit recognition. The MNIST database [46] contains size-normalized, center cropped 28×28 images of handwritten digits ranging from zero to nine. The objective of this task is to identify the intended digit given the handwritten version.

As input, the cerebellum simulator is given the intensity of each pixel from the image. Thus 784 Mossy Fibers are allocated to rate-encode pixel intensities. An additional 200 Mossy Fibers fire at high frequency regardless of input. As before, all state-encoding Mossy Fibers are randomized among the 2048 total Mossy Fibers.

Ten separate cerebellum simulators were trained, each with a single Microzone intended to recognize a single digit. (Memory limitations prevented a single simulator being trained with ten Microzones.) During training, the simulator was presented with an image for 500-milliseconds and allowed to rest for 500-milliseconds. If the image contained the digit that the simulator was being trained to identify, a single error was delivered to the Microzone at the end of the 500-ms viewing period. The simulator was trained for 1000 images, alternating between images containing the target digit and images containing a random digit.

At test time, each of the ten trained simulators was sequentially presented with the first 150 images of the test set, following the same paradigm of 500-ms viewing followed by 500-ms rest. The Microzone force 150-ms prior to the end of the viewing period for each digit was recorded, and the digit associated with the simulator maximizing force output at this time was taken as the overall prediction. Figure 13 shows the force output of several Microzones for the first ten MNIST test digits.

As Figure 14 indicates, the cerebellum achieved a MNIST precision score of 80%. This accuracy increases to 91% when considering the network's top two predictions. The confusion matrix indicates that the errors made by the network are qualitatively reasonable. For reference, a single layer artificial neural network achieved 88% precision on the same task [46] and requires significantly less computation. Nevertheless, cerebellar learning is evident and far better than random chance. This result adds evidence of the ability of the simulator to perform tasks featuring static pattern recognition with supervised error signals.

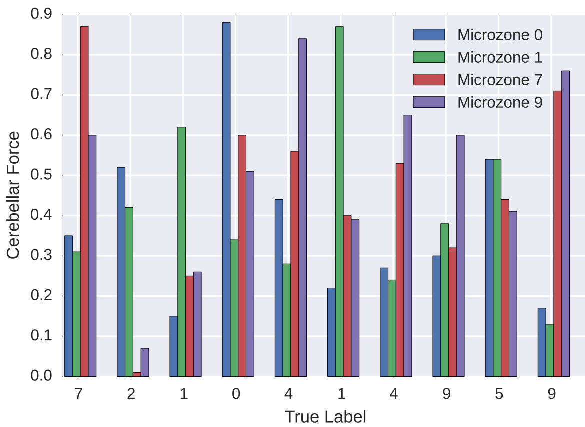
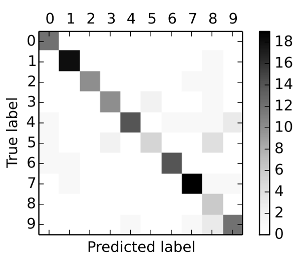


Fig. 13: Microzone activations for the first ten digits of the MNIST test set. From left to right, each cluster contains the activations of the Microzones intended to recognize digits 0, 1, 7, 9. True labels are given on the x-axis. Microzones forces are generally highest for the digit they are meant to recognize. However, some confusion may be seen in the high activations of Microzone 9 when seeing fours and sevens. Intuitively, these mistakes are reasonable as fours, nines, and sevens share common structure.



Accuracy	
Hit@1	0.80
Hit@2	0.91
Hit@3	0.95
Hit@4	0.97
Hit@5	0.99

Fig. 14: **Left:**MNIST confusion matrix indicates that the network is highly successful at recognizing ones, fours, and sevens. The largest missed predictions were fives and eights. These incorrect predictions could result from Microzone-5 being generally less forceful than Microzone-8. **Right:** MNIST Hit@k values indicate if the correct label was predicted within the top-k guesses.

10 Discussion

The previous sections explored two supervised control tasks: inverted pendulum balancing and autonomous vehicle control, two static pattern recognition tasks: boolean function and MNIST digits recognition, one temporal pattern recognition task, and one Reinforcement Learning task. Across all of these tasks, the same cerebellum simulator was used with the same number of cells, the same connectivity, and same learning rules, and the same parameters. Robust learning was observed in every domain when the cerebellum was provided with regularly occurring, supervised error signals. The two notable weak points were 1) weak discrimination between temporal patterns and 2) the inability to learn from delayed error signals. The former may highlight a limitation in the types of information granule cells are capable of recognizing. The latter stems from an inability of the simulated cerebellum to modulate the delay between its response to a stimulus and the incidence of an error signal (Figure 8). Even if the cerebellum were capable of modulating its force response to solve the credit assignment problem, any solution would result in it taking actions to minimize the incidence of future error signals. Doing so would only result in fewer error signals, which would lead to forgetting of learned behavior (Section 4.3) and at best, a policy that oscillates between correct and incorrect actions.

In general, the cerebellum simulator did not outperform tuned machine learning algorithms such as the PD-controller for autonomous vehicle control or the ANN for digit recognition. However, it did outperform online learners like Q-Learning and TEXPLORE. This trend was facilitated by the cerebellum simulator's ability to very quickly learn each new task - often reaching competitive performance within a few tens of episodes. Such an ability is crucial for humans who are faced with a diversity of tasks and a limited amount of time to collect experience.

11 Future Work and Conclusions

Combining the cerebellum simulator with other simulated brain regions could yield a more complete and capable model. Of particular interest is the basal ganglia, which is hypothesized to perform reinforcement learning [42]. If the combination of these two brain regions were able to handle delayed error signals, the set of learnable tasks could be greatly expanded. Additionally, the cerebral cortex is hypothesized to play a role in unsupervised learning [47]. The combination of these brain regions would encompass supervised, unsupervised, and reinforcement learning: the three known categories of machine learning.

Another question worth exploring is how cerebellar learning scales as a factor of the number of simulated cells. Eyelid Conditioning experiments indicated that the million cells used in this article improved fit with biological data compared to smaller models [30]. Perhaps even better performance be achieved by adding more cells. Unfortunately, changing the size of the simulation requires a labor intensive re-tuning of parameters, which has precluded exploration of different model sizes.

Can the cerebellum simulator be used as a function approximator? Albus [5] introduced the Cerebellar Model Articulation Controller (CMAC) function approximator which gave rise to *tile coding*, a state discretization method used commonly in reinforcement learning. However, the entire cerebellum could serve as a function approximator and for a reinforcement learning agent. Such a combination could ease the problems of temporal credit assignment inherent in the cerebellum simulator.

This article applies a biologically-plausible bottom-up simulation of the cerebellum to various control tasks. The simulation itself contains over a million cells and real time performance is achieved through GPU parallel processing. In previous work, the simulator has been tested on the eyelid conditioning task and found to successfully recreate animal data. This work further explores the learning capabilities and generality of the simulator by applying to the tasks of inverted pendulum balancing, autonomous vehicle control, robot balancing, static and sequential pattern recognition, and MNIST handwritten digit recognition. Overall, the cerebellum model is a supervised learner capable of handling a variety of tasks without reconfiguration so long as the error signals are regular and the state inputs do not require extended temporal pattern recognition. On the other hand, Reinforcement Learning proves problematic due to the delayed nature of error and the simulator's inability to solve the credit assignment problem. The cause of this problem is the inflexibility of the simulator to change the time delay between the incidence of the error signal and its own force response. In order to address the limitations of cerebellar learning, additional brain regions may need to be integrated. Nevertheless, the simulation proves to be a general and quick learner across many tasks explored in this article.

ACKNOWLEDGMENT

Special thanks to Patrick McAlpine for help with the 3D simulator. This work has taken place in the Learning Agents Research Group (LARG) at the Artificial Intelligence Laboratory, The University of Texas at Austin. LARG research is supported in part by grants from the National Science Foundation (CNS-1330072, CNS-1305287), ONR (21C184-01), AFRL (FA8750-14-1-0070), AFOSR (FA9550-14-1-0087), and Yujin Robot.

REFERENCES

[1] J. C. Eccles, M. Ito, and J. Szentgothai, "The mossy fiber input into the cerebellar cortex and its inhibitory control by golgi cells," in *The Cerebellum as a Neuronal Machine*. Springer Berlin Heidelberg, 1967.

[2] W. T. Thach, H. P. Goodkin, and J. G. Keating, "The cerebellum and the adaptive coordination of movement," *Annual Review of Neuroscience*, vol. 15, no. 1, pp. 403–442, 1992, pMID: 1575449.

[3] G. M. Shepherd, Ed., *The Synaptic Organization of the Brain*. Oxford: Oxford University Press, 1990.

[4] D. Marr, "A theory of cerebellar cortex," *The Journal of Physiology*, vol. 202, no. 2, pp. 437–470, Jun. 1969.

[5] J. S. Albus, "A theory of cerebellar function," *Mathematical Biosciences*, vol. 10, no. 1-2, pp. 25–61, Feb. 1971.

[6] M. Ito, "Long-term depression," *Annual Review of Neuroscience*, vol. 12, no. 1, pp. 85–102, 1989, pMID: 2648961.

[7] D. M. Wolpert, R. C. Miall, and M. Kawato, "Internal models in the cerebellum," *Trends in Cognitive Sciences*, vol. 2, no. 9, Sep. 1998.

[8] M. Kawato and H. Gomi, "The cerebellum and vor/okr learning models," *Trends in Neurosciences*, vol. 15, no. 11, pp. 445–453, 1992.

[9] J. C. Houk, "Cooperative control of limb movements by the motor cortex, brainstem and cerebellum," University of Massachusetts, Amherst, MA, Tech. Rep. COINS TR 89-118, Dec. 1989.

[10] J. C. Houk, J. T. Buckingham, and A. G. Barto, "Models of the cerebellum and motor learning," *Behavioral and Brain Sciences*, vol. 19, no. 3, pp. 368–383, 1996.

[11] A. Pellionisz and R. Llins, "Tensorial approach to the geometry of brain function: Cerebellar coordination via a metric tensor," *Neuroscience*, vol. 5, no. 7, pp. 1125–1136, 1980.

[12] A. Pellionisz and R. Llinas, "Tensor network theory of the metaorganization of functional geometries in the central nervous system," *Neuroscience*, vol. 2, no. 16, pp. 245–273, 1985.

[13] C. Hofstotter, M. Mintz, and P. F. M. J. Verschure, "The cerebellum in action: A simulation and robotics study," *European Journal of Neuroscience*, vol. 7, pp. 1361–76, 2002.

[14] N. R. Luque, J. A. Garrido, R. R. Carrillo, S. Tolu, and E. Ros, "Adaptive cerebellar spiking model embedded in the control loop: Context switching and robustness against noise," *Int. J. Neural Syst*, vol. 21, no. 5, pp. 385–401, 2011.

[15] C. Assad, S. Dastoor, S. Trujillo, and L. Xu, "Cerebellar dynamic state estimation for a biomorphic robot arm," in *SMC*. IEEE, 2005.

[16] R. E. Kettner, S. Mahamud, and A. G. Barto, "Prediction of complex two-dimensional trajectories by a cerebellar model of smooth pursuit eye movement." *Journal of Neurophysiology*, vol. 77, p. 2115, 1997.

[17] J. L. McKinstry, G. M. Edelman, and J. L. Krichmar, "A cerebellar model for predictive motor control tested in a brain-based device." *Proceedings of the National Academy of Sciences of the United States of America*, vol. 103, no. 9, pp. 3387–3392, Feb. 2006.

[18] J. L. McKinstry, A. K. Seth, G. M. Edelman, and J. L. Krichmar, "Embodied models of delayed neural responses: Spatiotemporal categorization and predictive motor control in brain based devices," *Neural Networks*, vol. 21, no. 4, pp. 553 – 561, 2008, robotics and Neuroscience.

[19] X. Ruan and J. Chen, "On-line nnac for a balancing two-wheeled robot using feedback-error-learning on the neurophysiological mechanism." *JCP*, vol. 6, no. 3, pp. 489–496, 2011.

[20] T. Yamazaki and J. Igarashi, "Realtime cerebellum: A large-scale spiking network model of the cerebellum that runs in realtime using a graphics processing unit," *Neural Networks*, vol. 47, pp. 103–111, 2013.

[21] J. C. Eccles, Ed., *The cerebellum as a neuronal machine*. Springer-Verlag, 1967.

[22] M. Ito, Ed., *The Cerebellum and Neural Control*. Raven Pr, 1984.

[23] J. Voogd and M. Glickstein, "The anatomy of the cerebellum," *Trends in Neurosciences*, vol. 21, no. 9, pp. 370–375, 1998.

[24] M. D. Mauk and N. H. Donegan, "A model of pavlovian eyelid conditioning based on the synaptic organization of the cerebellum." *Learning and Memory*, vol. 4, pp. 130–158, 1997.

[25] T. Ohyama, W. L. Nores, M. Murphy, and M. D. Mauk, "What the cerebellum computes," *Trends in Neurosciences*, vol. 26, 2003.

[26] J. F. Medina, K. S. Garcia, W. L. Nores, N. M. Taylor, and M. D. Mauk, "Timing mechanisms in the cerebellum: testing predictions of a large-scale computer simulation." *The Journal of neuroscience*, vol. 20, no. 14, pp. 5516–25, 2000.

[27] J. F. Medina and M. D. Mauk, "Computer simulation of cerebellar information processing," *Nature Neuroscience*, vol. 3, 2000.

[28] A. Pellionisz, R. Llinas, and D. H. Perkel, "A computer model of the cerebellar cortex of the frog," *Neuroscience*, vol. 2, pp. 19–36, 1977.

[29] 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, pp. 38–55, January 1994.

[30] W.-K. Li, M. J. Hausknecht, P. Stone, and M. D. Mauk, "Using a million cell simulation of the cerebellum: Network scaling and task generality," *Neural Networks*, vol. 47, pp. 95–102, November 2012.

[31] D. A. Mccormick and R. F. Thompson, "Cerebellum: essential involvement in the classically conditioned eyelid response," *Science*, vol. 223, no. 4633, pp. 296–299, Jan. 1984.

[32] C. Anderson, "Learning to control an inverted pendulum using neural networks," *Control Systems Magazine, IEEE*, vol. 9, no. 3, apr 1989.

[33] A. G. Barto, R. S. Sutton, and C. W. Anderson, *Neuronlike adaptive elements that can solve difficult learning control problems*. Piscataway, NJ, USA: IEEE Press, 1990, pp. 81–93.

[34] G. T. Kenyon, "A model of long-term memory storage in the cerebellar cortex: A possible role for plasticity at parallel fiber synapses onto stellate/basket interneurons," *Proceedings of the National Academy of Sciences USA*, vol. 94, p. 1420014205, 1997.

[35] J. S. Albus, *Brains, Behavior, and Robotics*. Peterborough: BYTE Books, 1981.

[36] K. H. Ang, G. C. Y. Chong, and Y. Li, "PID control system analysis, design, and technology," *Control Systems Technology*, Jul. 2005.

[37] P. Beeson, J. O'Quin, B. Gillan, T. Nimmagadda, M. Ristroph, D. Li, and P. Stone, "Multiagent interactions in urban driving," *Journal of Physical Agents*, vol. 2, no. 1, pp. 15–30, March 2008, special issue on Multi-Robot Systems.

[38] T. Hester and P. Stone, "TEXPLORE: real-time sample-efficient reinforcement learning for robots," *Machine Learning*, vol. 90, no. 3, pp. 385–429, 2013.

[39] T. Hester, M. Quinlan, and P. Stone, "RTMBA: A real-time model-based reinforcement learning architecture for robot control," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2012, pp. 85–90.

[40] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998.

[41] C. Ghez and S. Fahn, "The cerebellum," *Principles of Neural Science*, vol. 2, 1985.

[42] K. Doya, "Complementary roles of basal ganglia and cerebellum in learning and motor control," *Current Opinion in Neurobiology*, vol. 10, no. 6, pp. 732–739, 2000.

[43] L. E. Baum and T. Petrie, "Statistical inference for probabilistic functions of finite state Markov chains," *Annals of Mathematical Statistics*, vol. 37, pp. 1554–1563, 1966.

[44] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proc. 18th International Conf. on Machine Learning*. Morgan Kaufmann, San Francisco, CA, 2001, pp. 282–289.

[45] B. E. Kalmbach, H. Voicu, T. Ohyama, and M. D. Mauk, "A subtraction mechanism of temporal coding in cerebellum," *Journal of Neuroscience*, vol. 31, pp. 2025–2034, 2011.

[46] Y. L. L. Cun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[47] K. Doya, "What are the computations of the cerebellum the basal ganglia and the cerebral cortex?" *Neural Networks*, vol. 12, no. 7-8, 1999.

[48] W. Li, "Timing in the Cerebellum: A Matter of Network Inhibition," Ph.D. dissertation, University of Texas Austin, May 2015.

Appendix A
Simulator Equations and Parameters

This appendix highlights selected components of the simulated cerebellum. A comprehensive treatment is given in [48]. Cellular regions are abbreviated as follows: Mossy Fibers (MF), Golgi Cells (GO), Granule Cells (GR), Parallel Fibers (PF), Stellate Cells (SC), Basket Cells (BF), Purkinje Cells (PC), Inferior Olive (IO), Nucleus Cells (NC), Climbing Fibers (CF).

A.1 Representation of Neurons

Simulated neurons are implemented using a single compartment leaky integrate and fire representation [29], [27]. This neuron model has the advantages of simplicity and computational tractability, allowing over a million interconnected neurons to be simulated in parallel. The leaky integrate and fire neuron spikes when the membrane potential V_m exceeds a threshold h. After firing, neuron's threshold increases to h_{max} to emulate the absolute and relative refractory periods. After these spike-initiated increases, the threshold decays exponentially (h_{decay}) back to its normal level h_{base}:

$$\text{Spiking} = V_m > h_t \tag{4}$$

$$h_t = \begin{cases} h_{max} & \text{if Spiking} \\ h_{t-1} - (h_{t-1} - h_{base}) \cdot h_{decay} & \text{otherwise} \end{cases} \tag{5}$$

Membrane potential V_m is calculated from synaptic current I_{syn}, leak conductances E_l, and membrane capacitance C as follows:

$$\frac{dV_m}{dt} = \frac{-g_l \cdot (V_m - E_l) - \sum_{n=0}^{synapses} I_{syn}^{n}(V_m, t)}{C} \tag{6}$$

The first term represents the contribution from the leak conductance to the change in membrane potential and the second term sums over all different synapses contacting the postsynaptic cell. These parameters are modeled on known physiological data for each cell type and are provided in Table 2. After fine-tuning each neuron type to match published physiological data, the leaky integrate and fire model yields representations that are both computationally efficient and accurate.

A.2 Simulation of Synaptic Potentials

Transmission of information between connected neurons is facilitated by the movement of charge through the synapse connecting the neurons. This synaptic current is given by:

$$I_{syn}(V_m, t) = \bar{g}_{syn} \cdot g_{syn}(t) \cdot (V_m(t) - E_{syn}) \tag{7}$$

$\bar{g}_{syn}$ scales synaptic strength, E_{syn} is the synaptic reversal potential, and $g_{syn}(t)$ gives the time course of the underlying conductance as expressed by:

$$\frac{\mathrm{d}g_{syn}}{\mathrm{d}t} = \sum_{i=0}^{inputs} S_i \cdot w_i \cdot (1 - g_{syn}) - g_{syn}\tau_{syn} \tag{8}$$

This summation steps through all presynaptic inputs. S_i represents a spike in the ith presynaptic input, w_i is the synaptic weight of the ith presynaptic input, and τ_{syn} is the decay time constant for the synaptic potential. Thus, synaptic currents were simulated with an instantaneous rise and an exponential decay by summing all inputs of a particular type into a single current that saturates at 1.0 and decays at the rate of τ. Specific values for τ were chosen on the basis of the wealth of electrophysiological data that exists for cerebellar synapses and are provided in Table 2.

A.3 Synaptic Plasticity

The cerebellum simulation models two sites of synaptic plasticity, the synapses between the granule and Purkinje cells (GR_i) and the synapses between the mossy fibers and deep nuclei (MF_i). Plasticity at these two sites is calculated as follows:

$$\Delta w_i^{gr} = \delta_-^{gr}\mathrm{GR}_i\mathrm{CF}(100) + \delta_+^{gr}\mathrm{GR}_i(1 - \mathrm{CF}(100)) \tag{9}$$

$$\Delta w_i^{mf} = \delta_-^{mf}\mathrm{MF}_i\Theta_{LTD}^{PKJ}(50) + \delta_+^{mf}\mathrm{MF}_i\Theta_{LTP}^{PKJ}(50) \tag{10}$$

$\delta_-^{gr} = -0.0027$ and $\delta_+^{gr} = 0.0003$ represent the magnitude of the step decreases and increases in the synaptic weight. CF(100) is 1 for the 100ms after a climbing fiber spike and 0 otherwise. GR_i is 1 whenever the ith granule cell has fired prior to CF(100) window and 0 otherwise. In Equation 10 $\delta_-^{mf} = -0.0000125$ and $\delta_+^{mf} = 0.00003$ are constants that represent the magnitude of the step decreases and increases in the synaptic weight. MF_i is 1 whenever the ith mossy fiber fires and 0 otherwise. $\Theta_{LTD}^{PKJ}(50)$ is 1 whenever the average Purkinje cell activity seen in the preceding 50 msec by the postsynaptic nucleus cell increases over a threshold value (∼80hz). Similarly, $\Theta_{LTP}^{PKJ}(50)$ equals 0 except when the average Purkinje cell activity falls below a threshold value (∼40hz). Synaptic weights were restricted to the interval [0,1] by preventing further changes in the same direction when the synaptic weights reached 0 or 1. The parameters δ_-^{gr}, δ_+^{gr}, δ_-^{mf}, and δ_+^{mf} were all initially tuned to maximize fit with known biological data for spontaneous activity of cerebellar neurons and rate of learning in the eyelid conditioning task. On the inverted pendulum domain δ_-^{gr} and δ_+^{gr} were increased to $-0.045, 0.005$ respectively to speed up the learning rate.

h	GO	GR	PF	SC	BC	PC	IO	NC
h_{max}	-10	-20	-48	0	10	-48	-61	5
h_{base}	-33	-40	-60	-50	-50	-60	10	-40
h_{decay}	20	3	5	22	0	5	122	-72
E_l	-70	-70	-	-60	-70	-60	-60	-65
g_l	.02	.1	-	.2	.2	.2	.15	.1

E_{syn}	GR	PC	BC	NC	IO
MF	0	-	-	-	-
GO	-80	-	-	-	-
PC	-	-	-70	-80	-
SC	-	-80	-	-	-
BC	-	-80	-	-	-
NC	-	-	-	-	-80

$\bar{g}_{syn}$	GO	GR	SC
MF	.0049	.00374	-
GO	.00784	.022	-
GR	8.75e-5	-	-
PF	-	-	.00132

τ_{syn}	GO	GR	PC	SC	BC	NC
MF	4.5	55	-	-	-	-
GO	25	25	-	-	-	-
GR	4.5	-	-	-	-	-
PF	-	-	4.15	4.15	4.15	-
PC	-	-	-	-	5	4.15
SC	-	-	4.15	-	-	-
BC	-	-	5	-	-	-

TABLE 2: From top to bottom: synaptic threshold h-values, leak conductances E_l and conductance scaling g_l (Equations 6-5), synaptic reversal potentials E_{syn} (Equation 7), synaptic gains $\bar{g}_{syn}$ (Equation 7), and synaptic τ-values (Equation 8). Data is row-major, e.g. τ_{syn}MF→GR = 55.