A REAL TIME ONLINE CONTROL SYSTEM USING A CEREBELLUM CIRCUIT SIMULATED ON NEUROMORPHIC HARDWARE

A PREPRINT

First author **Second author**

September 2, 2020

ABSTRACT

This work presents the implementation of a biologically inspired and adaptive robot control system on neuromorphic hardware. The control system is built from spiking neurons organised in a structure inspired by the cerebellum. The network learns to perform the vestibulo-ocular reflex (VOR) in real time. The system is shown to perform the task in simulated and physical environments.

***K*eywords** Cerebellum · robot control system · spiking neurons · neuromorphic computing · SpiNNaker

1 Introduction

Classical control theory has produced robot control systems capable of accurate and complex manipulation with wide-spread industrial adoption. Indeed this has been made possible by targetting specific, narrow applications and efforts now exist to increase the breadth of domains in which single controllers can operate nominally.

Brains have had billions of years to perfect motor control for various body morphologies, material choices, environments and uses. Thus, they could be seen as gold standards to be matched or outdone by human engineered solution. One of the systems involved in motor control and adaption is the cerebellum.

We want to use SpiNNaker's acceleration of SNNs to implement a biologically inspired cerebellum that learns in a biologically inspired way to perform the Vestibulo-ocular reflex (VOR) in real time. This work uses a model designed by Naveros et al. [7]. The model is ported onto SpiNNaker [4, 5, 3] and simulated using sPyNNaker [8]. The circuit's ability is shown in a tightly-coupled simulated environment, in a loosely-coupled simulated environment (NRP) and in a physical robot.

SpiNNaker's scalability can then be exploited to employ copies of the same network to solve control tasks which involve more than 2 degrees of freedom.

Introduction covers robot control systems using SNNs: Biological plausibility is not the only criterion for using SNNs. We also want to accelerate very complex simulations to perform complex tasks. SNN controller on neuromorphic hardware could outperform traditional controllers: [1].

[2] other real time cerebellum with learning.

More cerebellum control system [6]

More cerebellum control experiments run on other platforms: CPUs, GPUs, NPUs

1.1 Contribution

We implement a cerebellum model as an adaptive control system on SpiNNaker, thus allowing to be run in real time. We validate it using a simulated environment and with a physical robot. The simulated environment running directly on SpiNNaker, called SpiNNGym, allows for environments to be simulated in a tightly-coupled fashion to the development of neural circuits.

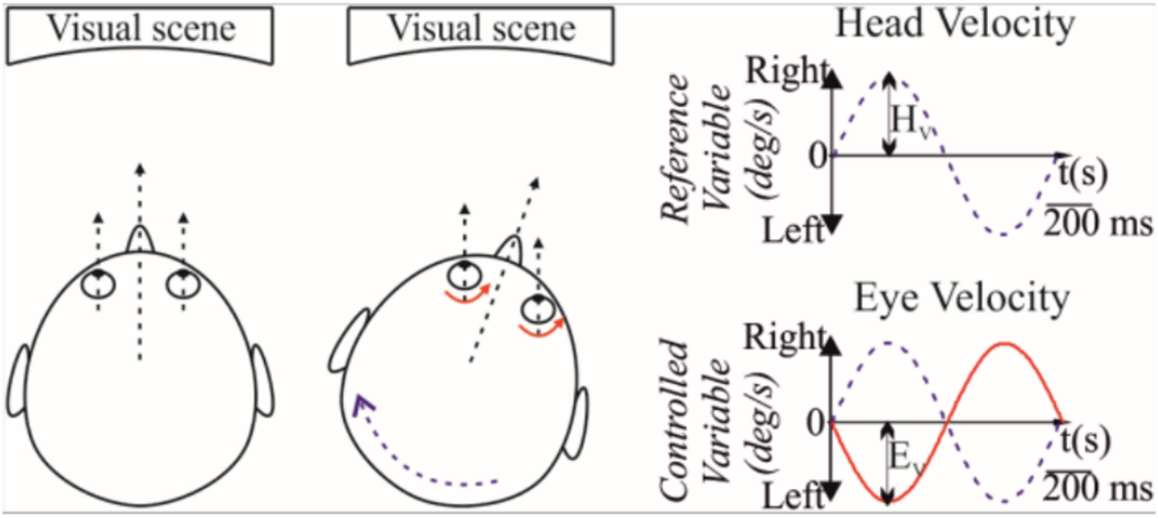

Figure 1. Vestibulo-ocular reflex (VOR) experiment. VOR stabilizes the visual field during horizontal head rotation tests (r-VOR) by producing contralateral eye movements.

Figure 1

2 Methods

2.1 Network architecture

The network proper consists of 4 populations or groups of neurons. The excitatory populations are the granule cells (GrC) and the Vestibular Nuclei cells (VN). The inhibitory populations are the Golgi cells (GoC) and Purkinje cells (PC).

In addition to these populations of spiking neurons there are 2 populations of non-biological neurons that produce spikes at pre-defined times. These input populations are the Mossy Fibers (MF, excitatory) and Climbing Fibers (CF, training). The MF produce the stimulus activity presented to GrC, GoC and VN encoding the head position (using the first 50 neurons) and velocity (using the last 50 neurons). The CFs produce spikes seen solely by PCs encoding the error between the head and eye position.

A further 2 non-spiking (agonist and antagonist) neurons are used to control the direction of eye movement based on the activity of the VN cells. These neurons simply accumulate spiking activity over time, with the one having recorded the largest number of spikes at sample time deciding the direction of eye movement. The relative difference in firing rate influences the magnitude of the eye velocity.

The architecture is presented in Figure 2.

Explain how this scenario can 1. be augmented to control MF as well 2. how this type of structure would work with a physical icub or NRP implementation.

2.2 Stimulus generation

The stimulus produced by MFs is periodic: a trial of a rotary VOR stimulation lasts 1 second. It encodes the position of the head between its minimum (all the way left) and maximum (all the way right) positions (Figure 3). Neuron space is split in half, with the first half (50 neurons) encoding the head position, while the second half (50 neurons) encodes the head velocity. Individual neurons within MF spike according to a Poisson process with one of two rates: a low rate (0 Hz) or a high rate (600 Hz). The head position and velocity are represented using the same method. The space of

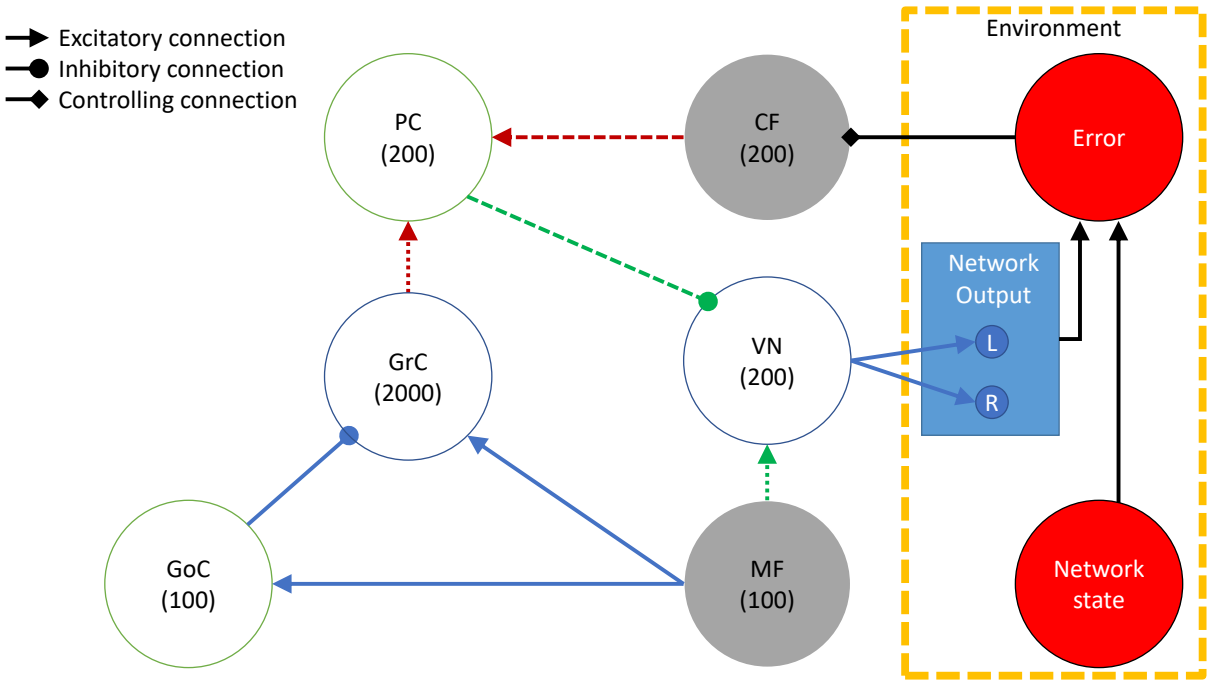

Figure 2: Cerebellum-inspired network architecture. The elements contained within the yellow dotted rectangle represent the external elements to the cerebellar circuit. This rectangle is represented by the SpiNNGym environment for simulated experiments and the iCub robot for physical experiments.

neurons is covered by Gaussian receptive fields with a $\sigma = 0.02$ and mean centred on the current value of position or velocity. This results in at most 8 neurons producing spikes at a rate of 1 Hz or higher each for position and velocity.

`https://github.com/EduardoRosLab/VOR_in_neurorobotics/blob/master/error_activity.py`

CFs encode the error between the head and eye positions. Individual neurons within CF spike according to a Poisson process with one of two rates: a low rate (2 Hz, c.f. 1 Hz in [7]) or a high rate (20 Hz, c.f. 10 Hz in [7]). A perfect head velocity compensation would see the eye velocity signal in counter-phase with equal amplitude. CFs are also divide into 2 equal subsets of neurons: one representing the error of the agonist response and the other representing the error of the antagonist response. A single error signal is computed that combines position and velocity errors:

$$e = 0.1e_p + 0.9e_v \tag{1}$$

where e is the global combined error, e_p is the positional error and e_v is the velocity error.

How to select which output neurons produce high or low error?

An error is 0 is produced if the eye position and velocity are in anti-phase with the respective head quantities.

For positive errors, the antagonist response needs to be adjusted. Conversely, negative errors lead to the agonist response being adjusted.

`https://github.com/EduardoRosLab/VOR_in_neurorobotics/blob/master/eye_twist.py`

The L/R controller neurons are sampled every 10 ms (a sampling frequency of 100 Hz). When sampled, the neuron with the highest sub-threshold potential is the "winner" resulting in an eye movement in the decided direction (either left or right). These neurons each receive connections from VN, with one receiving the agonist response and the other receiving the antagonist response. The eye velocity is compute according to Equation 2:

$$v_e = \frac{2N\pi * 1.55}{M}(l_r - r_r) \tag{2}$$

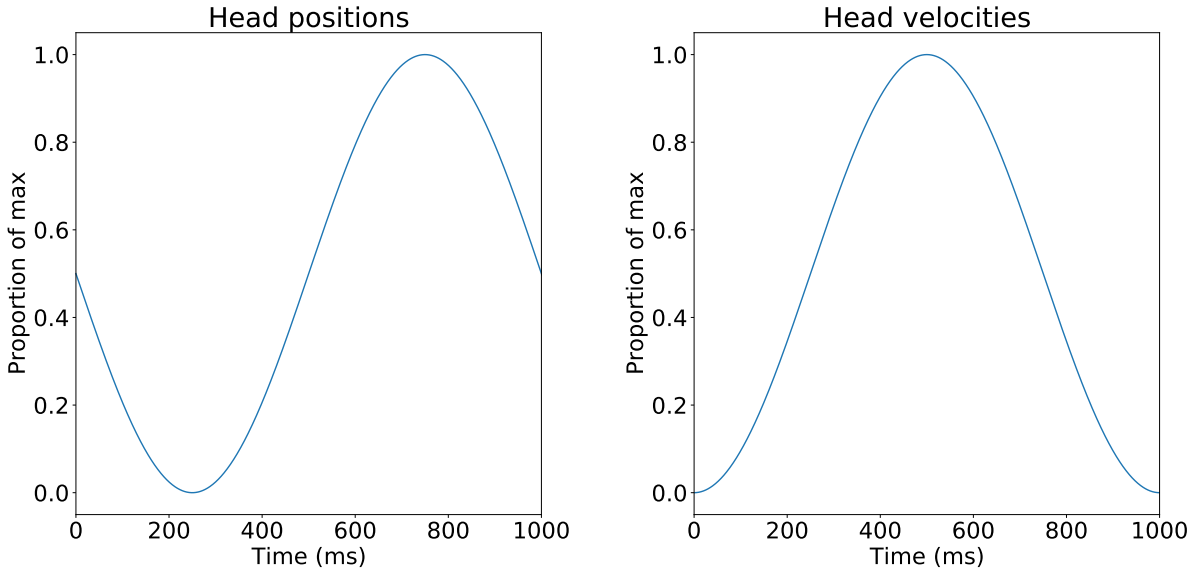

Figure 3: Periodic head position and velocity in a trial.

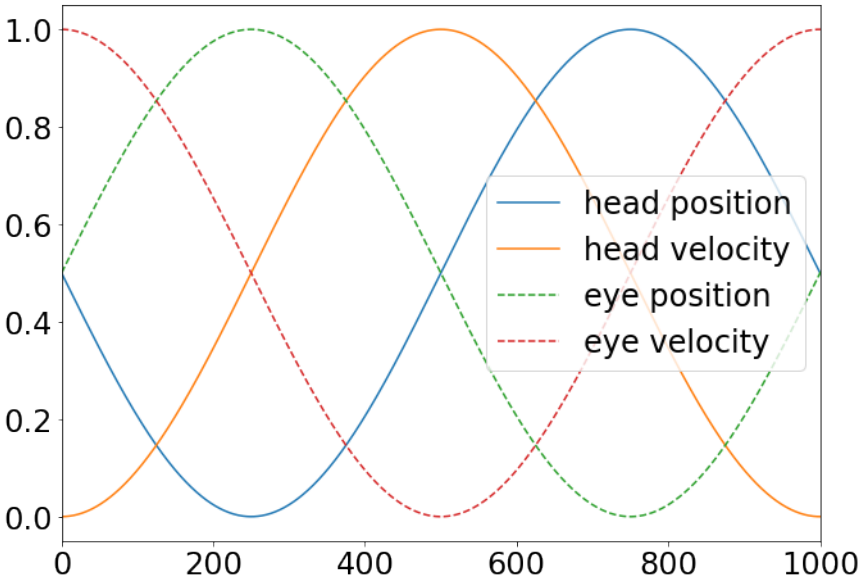

Figure 4: Perfect eye position and velocity in relation to head position and velocity.

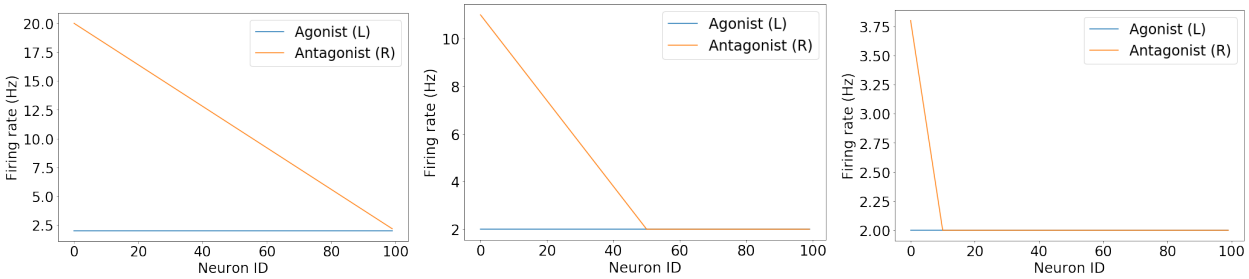

Figure 5: Example firing rates set for each neuron in the agonist and antagonist

Connection name	# of synapses	Initial Weight (nA)	Delay (ms)
MF–GrC	40000	$0.5 \pm 20\%$ (uniform)	1-10 ms (uniform)
MF–GoC	100	$0.1 \pm 20\%$ (uniform)	1-10 ms (uniform)
MF-VN	20000	0.0005	1-10 ms (uniform)
GrC–PC	400000	0.005	1-10 ms (uniform)
CF–PC	200	N/A	1 ms (teaching signal)
GoC–GrC	16000	0.005	1-10 ms (uniform)
PC–VN	200	0.01	1 ms (teaching signal) or 1-10 ms (uniform)

Table 1: Connection parameters. Connections are ordered first by their afferent, pre-synaptic population, then by their efferent, post-synaptic population.

where v_e is the velocity imparted to the eye, N is the previously defined normalisation factor based on the maximum amplitude of the signal, $M = 100 * 85$ for unknown reasons and with unknown function, l_r is the agonist (left) firing rate and r_r is the antagonist (right) firing rate.

2.3 SpiNNGym

Using SpiNNGym, it is possible to record the outgoing (from environment to neural circuit) spiking activity, number of spikes received per 10 ms for L/R controller neurons as well as the value of the error computed per 10 ms. The behaviour of the SpiNNGym is deterministic through the appropriate use of RNG seeds for the spike generation of MF. MF activity is pre-defined and periodic. MF is a variable rate (inhomogeneous) Poisson spike source with rates that change every 10 ms.

L/R controller neurons are spike counters.

CF is computed based on L/R controller neurons and current position and velocity. Current position and velocity need to be tracked throughout the simulation. These can be pre-generated and loaded onto the vertex. CF neurons are Poisson spike sources with rates computed every 10 ms based on the previous 10 ms of activity.

I think it would be a good idea for the trial length (from 1 s to e.g. 2 s) to be variable within SpiNNGym. If we want to go from simulation to physical robot, it might be a good idea to train the network on different head speeds for better **transfer learning**.

2.4 Learning rules

2.5 Experimental parameters

All spiking neurons in the network use the same cell and synapse parameters: cell membrane capacitance (CM) = 1 nF, IC = 0 nA, $\tau_m = 20$ ms, $\tau_{refract} = 0.1$ ms, $\tau_{syn_E} = 5$ ms, $\tau_{syn_I} = 5$ ms, $v_{reset} = -65$ mV, $v_{rest} = -70$ mV, $v_{thresh} = -50$ mV.

Q: in the original paper, CF–PC has a weight of 0.005. Are we meant to have a weight on that connection too?

2.6 Experiments

- replicate results from the original paper
- does the learning deteriorate if the number of learning steps is greatly increased?

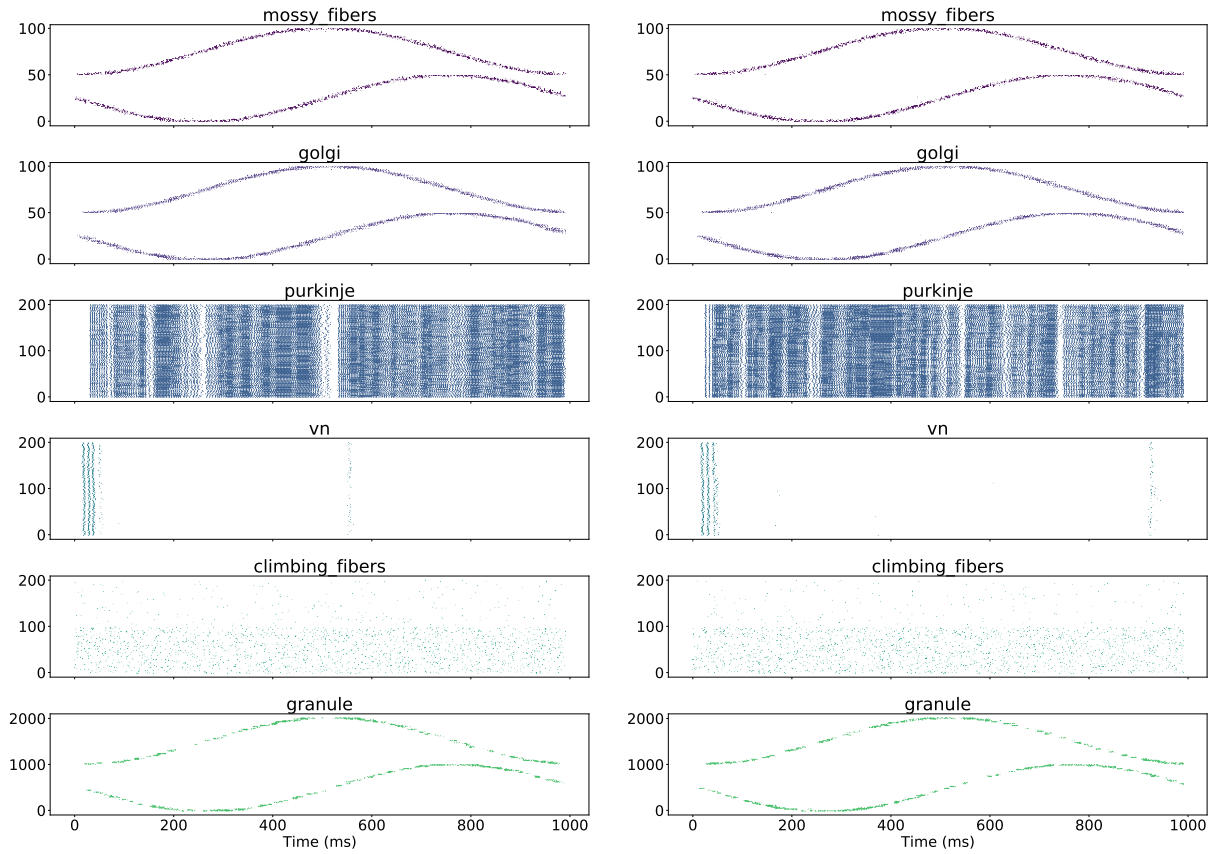

Figure 6: Spike raster for all populations in the network for a 1 s trial. LEFT: Original. RIGHT: Master.

3 Results

3.1 Activity validation

3.2 Single PC experiments

Original weight: 0.10000038

Master weight: 0.03624725

3.3 Single VN experiments

Original weight increases: 0.01500511 0.02500534 0.03500557 0.0450058 0.05500603 0.06500626 0.07500648 0.08500671 0.09500694 0.10000038

Master weights increases: 0.01062393 0.02062416 0.03062439 0.04062462 0.05062485 0.06062508 0.07062531 0.08062553 0.09062576 0.10000038

3.4 Results to match

4 Discussion

References

[1] Ignacio Abadia, Francisco Naveros, Jesus A. Garrido, Eduardo Ros, and Niceto R. Luque. On Robot Compliance: A Cerebellar Control Approach. *IEEE Transactions on Cybe rnetics*, pages 1–14, 2019. ISSN 2168-2267. doi: 10.1109/tcyb.2019.2945498.

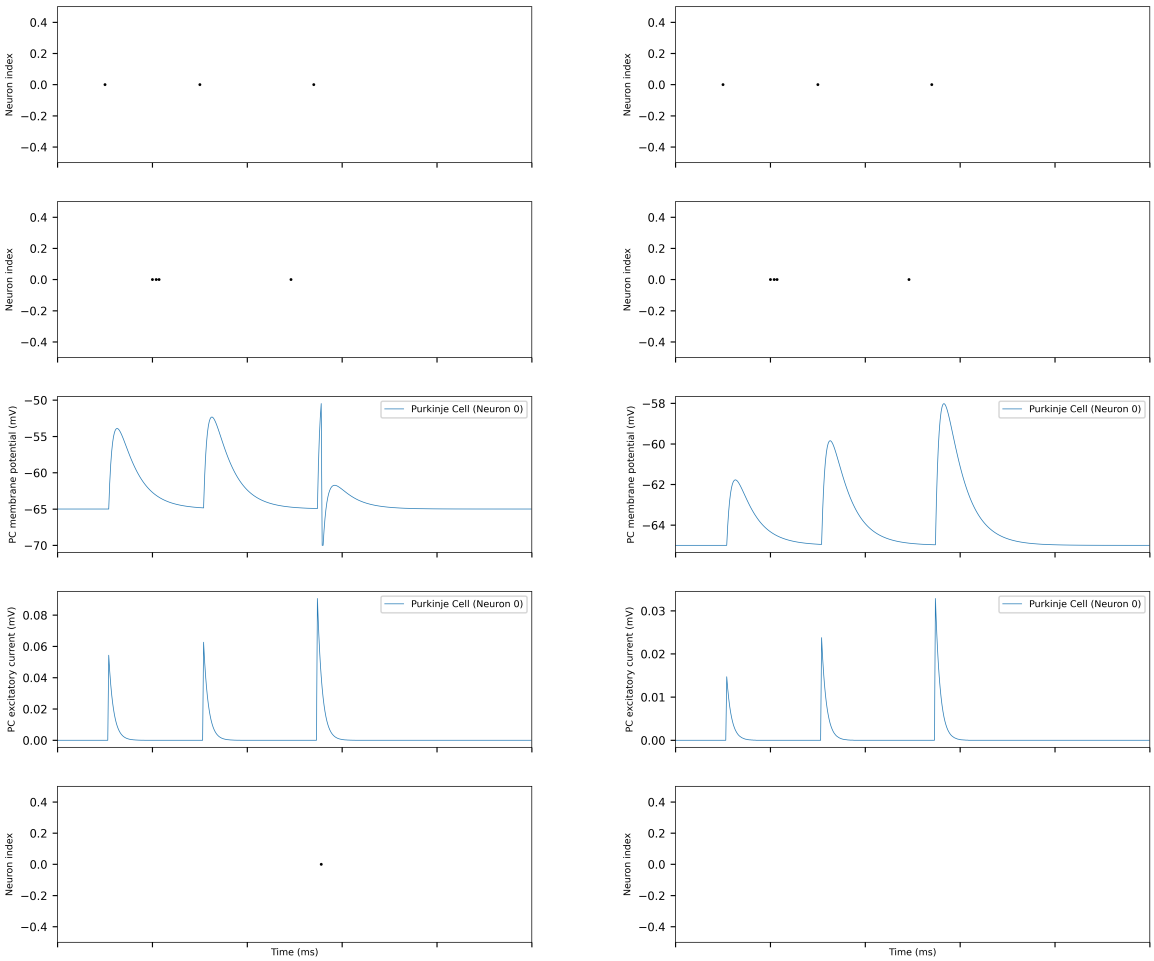

Figure 7: Results with the most recent version of the tools. LEFT: Original. RIGHT: Master.

[2] Richard R. Carrillo, Eduardo Ros, Christian Boucheny, and Olivier J.M.D. Coenen. A real-time spiking cerebellum model for learning robot control. *BioSystems*, 94(1-2):18–27, 2008. ISSN 03032647. doi: 10.1016/j.biosystems.2008.05.008.

[3] Steve Furber and Petruț Antoniu Bogdan, editors. *SpiNNaker: A Spiking Neural Network Architecture*. now publishers, Boston-Delft, 1 edition, 2020. ISBN 978-1-68083-653-0. doi: 10.1561/9781680836530. URL `https://nowpublishers.com/article/BookDetails/9781680836523`.

[4] Steve B Furber, David R. Lester, Luis A. Plana, Jim D. Garside, Eustace Painkras, Steve Temple, and Andrew D. Brown. Overview of the SpiNNaker system architecture. *IEEE Transactions on Computers*, 62(12):2454–2467, 2013. ISSN 00189340. doi: 10.1109/TC.2012.142.

[5] Steve B Furber, Francesco Galluppi, Steve Temple, and Luis A. Plana. The SpiNNaker project. *Proceedings of the IEEE*, 102(5):652–665, 2014. ISSN 00189219. doi: 10.1109/JPROC.2014.2304638.

[6] Niceto Rafael Luque, Jesús Alberto Garrido, Richard Rafael Carrillo, Olivier J.M.D. Coenen, and Eduardo Ros. Cerebellarlike corrective model inference engine for manipulation tasks. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 41(5):1299–1312, 2011. ISSN 10834419. doi: 10.1109/TSMCB.2011.2138693.

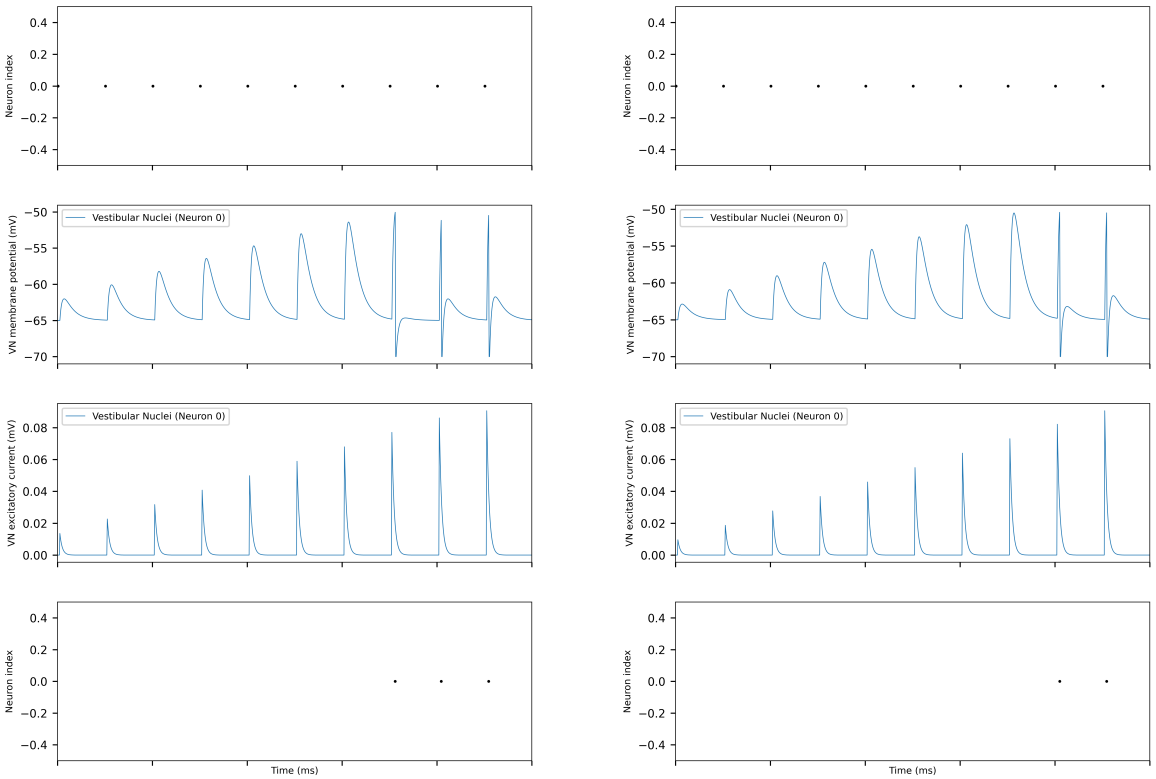

Figure 8: Results with the most recent version of the tools. LEFT: Original. RIGHT: Master.

[7] Francisco Naveros, Jesus A. Garrido, Angelo Arleo, Eduardo Ros, and Niceto R. Luque. Exploring Vestibulo-Ocular Adaptation in a Closed-Loop Neuro-Robotic Experiment Using STDP. A Simulation Study. *IEEE International Conference on Intelligent Robots and Systems*, pages 6706–6711, 2018. ISSN 21530866. doi: 10.1109/IROS.2018.8594019.

[8] Oliver Rhodes, Petruț A Bogdan, Christian Brenninkmeijer, Simon Davidson, Donal Fellows, Andrew Gait, David R Lester, Mantas Mikaitis, Luis A Plana, Andrew G D Rowley, Alan B Stokes, and Steve B Furber. sPyNNaker : A Software Package for Running PyNN Simulations on SpiNNaker. *Frontiers in Neuroscience*, 12(November), 2018. ISSN 1662-453X. doi: 10.3389/fnins.2018.00816.

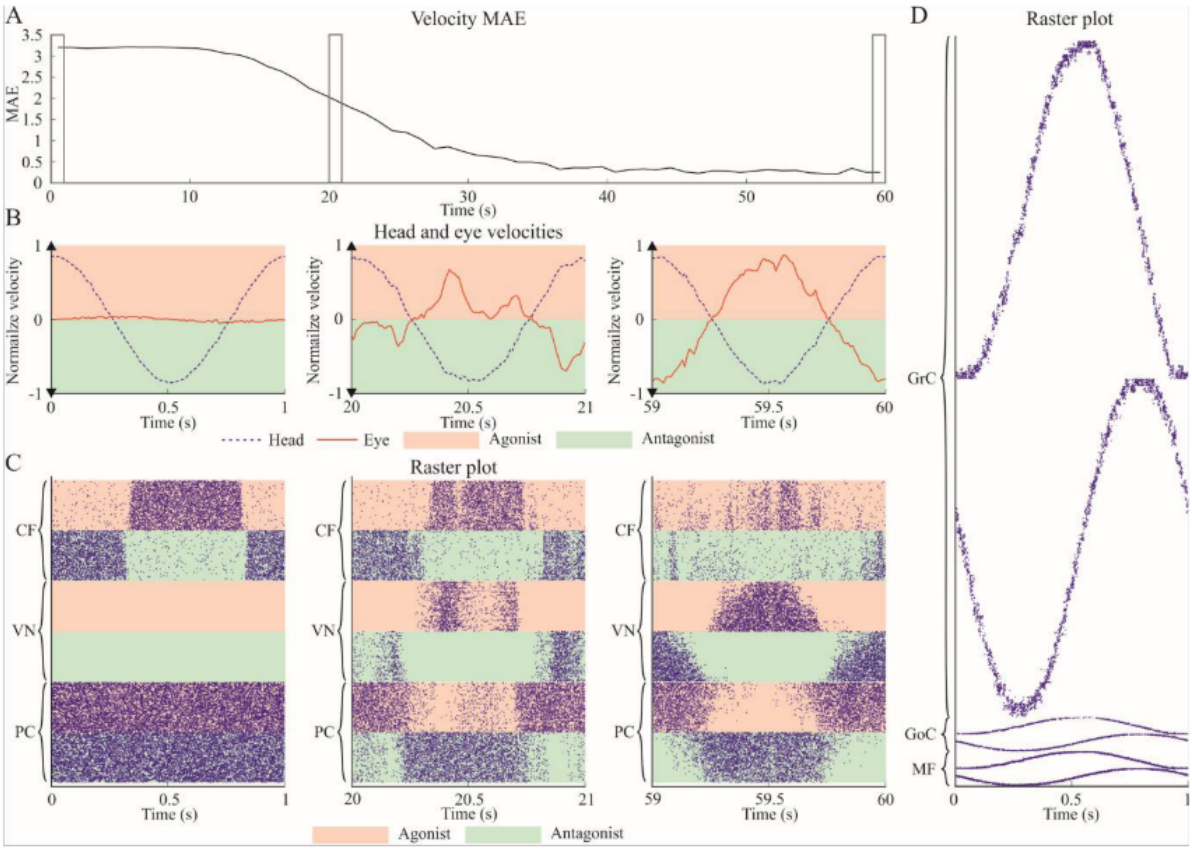

Figure 3. iCub robot signals and cerebellar neural activity in a r-VOR task. A) Plot depicting the Mean Absolute Error (MAE) between head and eye velocities that are obtained for each trial during the r-VOR cerebellar adaptation process. For a perfect cerebellar head velocity compensation, the eye velocity signal must be in counter-phase with equal amplitude. B) Head and eye velocity trajectories at three different learning stages; initial, middle and final cerebellar adaptation process. C) CF, VN and PC neural activity at three different learning stages: initial, middle and final cerebellar adaptation process. The neural activity of these layers evolves with the cerebellar adaptation mechanisms. The CF-PC-VN cerebellar sub-circuitry is divided in two antagonist micro-complexes responsible for head velocity compensation in clockwise and anticlockwise direction. D) MF, GoC and GrC neural activity. The vestibular signals produced as consequence of the head velocity remain unchanged as well as the neural activity caused by these vestibular signals.

Figure 9