Learning Anticipatory Behaviour Using a Simple Cerebellar Model

Harri Valpola[⋆]

[⋆]Laboratory of Computational Engineering
Helsinki University of Technology
P.O.Box 9203, FI-02015 TKK
Harri.Valpola@tkk.fi

Abstract

The cerebellar system is one of the best known components in the brain. Its computational principles are understood in sufficient detail to allow for using them for engineering applications. In this paper, some of the known facts about the cerebellar system are reviewed. The computational task of the cerebellar system appears to be prediction. On the other hand, the cerebellar system participates in motor control tasks. Feedback-error learning is one setting where a controller can be utilized in motor control. The limits of achievable temporal accuracy in feedback-error learning are investigated experimentally and it is shown that—somewhat surprisingly—motor output can have better temporal resolution than the error signal which is used for adapting the predictive controller. Finally, the algorithm is demonstrated in a simple but demanding simulated robot-control task.

1 Introduction

We still have plenty to learn from the brain. We would like to understand the underlying “algorithms” and computational principles and implement them in our artifacts. Not all details will be relevant but it can be expected that some general principles will turn out to be applicable, just as wings turned out to be a useful feature in airplanes but flapping was neither compatible with our technology nor necessary for flying.

Muscles are the major output of the brain (in addition to some hormonal regulation) and it is therefore fair to say that the brain has evolved to control movement. In order to understand the brain at a system-level, it is therefore useful to consider the problems encountered when controlling a body and interacting with the environment.

In this paper I will focus on anticipatory motor control implemented by the cerebellar system. It is one of the best understood parts of the brain both in terms of neurophysiology and computational principles. However, it seems that while this knowledge exists, it is not widespread because much of neuroscience research is focusing on neocortex in general and sensory processing in particular. Largely this is because it is easier to make well-controlled experimental setups but also because cortex is responsible for higher cognitive functions like planning, reasoning, attention and consciousness. Nevertheless, even in order to understand the neocortex, it is important to maintain a broader picture of the brain: there are many things that the neocortex does *not* need to do because other systems, like cerebellum, are taking care of them. The aim of this paper is to explain in simple terms what the cerebellar system does and demonstrate this in simulations.

2 Self-supervised learning

Thanks to extensive experimental, theoretical and engineering research, the cerebellar “algorithm” is nowadays well understood (see,e.g., Marr, 1969; Albus, 1971; Kawato and Gomi, 1992; Smith, 1998; Barto et al., 1999; Medina et al., 2000b,a; Medina and Mauk, 2000; Hansel et al., 2001; Hofstötter et al., 2002; Garwicz, 2002; Ito, 2002; Ohyama et al., 2003; Daniel and Crepel, 2003; Grethe and Thompson, 2003; Kawato, 2003; Apps and Garwicz, 2005; McKinstry et al., 2006). A brief summary of the research is that the cerebellar system turns out to be a predictor. In machine learning, predictors are usually adapted using supervised learning: there is a teacher who provides the correct answers and the task is to learn to mimic the answers provided by the teacher. Prediction tasks are about the only sensible supervised learning tasks in autonomous systems since there is no point of reinventing the wheel: if the teacher needs to be implemented anyway, there is no benefit in replicating the behaviour of the teacher. Supervised learning of prediction does make sense for an autonomous system because the system learns to provide the answers *before* the teacher gives the answers. Also, supervised learning is possible with a teacher who cannot give the correct answer but can give a correction. The student (or adaptive system) can therefore give faster and better answers than the teacher. This is pretty much the role of the cerebellar system: *the cerebellar system is specialized in making accurately-timed predictions in the timescale of about 100 ms – 5 s*.

Since the teachers are also located in the brain, one can talk about *self*-supervised learning. Cerebellar predictions are used for many purposes and correspondingly there are many types of teachers. In motor tasks, the teacher can be, for instance, a spinal or brain-stem reflex or it can be a focused, conscious decision of the neocortex. Cerebellum is implicated not only in motor tasks but in purely sensory and cognitive tasks, too (Allen et al., 1997). It is true not only for the cerebellar system but for many other adaptive modules in the brain that it is not the task but the algorithm that provides the most natural basis for understanding the modules. Nevertheless, motor tasks seem to be the most typical ones for the cerebellar system and they are also the focus in this paper.

3 Cerebellar system

Before going further into motor control, I will briefly introduce the anatomy of the cerebellar system and link it with the algorithm discussed above.

The cerebellar system consists of the inferior olive nucleus (IO), cerebellar cortex (CB) and deep nuclei (DN). Their computational roles are depicted in Figure 1. The teaching signals are relayed through IO (but are generated by diverse systems as noted earlier). The climbing fibres from IO deliver the teaching signal to CB and DN which implement the input-output mapping of the cerebellar system.

Roughly speaking, the cerebellar cortex is responsible for the accurate timing of cerebellar predictions but the deep nuclei implement the (linear) association between the inputs and outputs. Timing is implemented as disinhibition: normally CB inhibits DN but CB releases its inhibition at just the right moment to allow DN to activate its outputs (Perrett et al., 1993; Medina et al., 2000b). Note that when people talk about cerebellum, they usually refer to the cerebellar cortex, shown in Figure 2. This is because it is

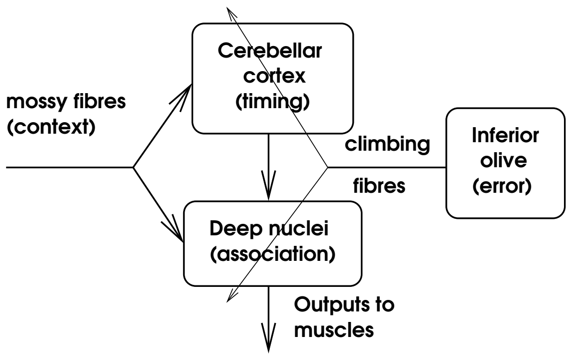

Figure 1: The "standard" architecture of the cerebellar system. In the part of cerebellum responsible for eye-movements, deep nuclei are missing from the loop and the cerebellar cortex is directly controlling brain stem.

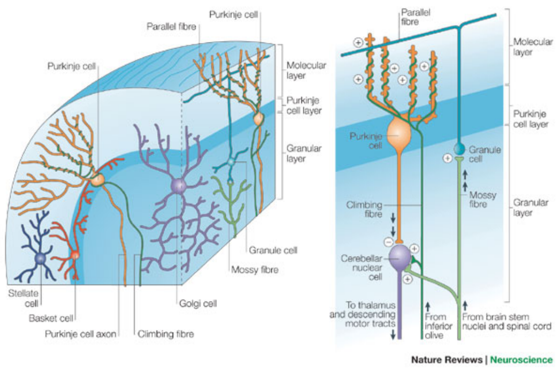

Figure 2: The anatomical structure and connections of the cerebellar cortex. The figure is reprocuded from (Apps and Garwicz, 2005).

anatomically much larger and most of the computations seems to be going on there.

4 Feedback-error learning and unknown delays

The "cerebellar algorithm" is supervised prediction which has no direct links with motor control and there are many ways in which a predictor could be used in motor control. Looking at the cerebellar system alone therefore does not tell much about the role cerebellar system plays in motor control. In fact, it is quite possible that there are multiple different uses. I will illustrate this with a gaze stabilization task.

When the visual images moves on the retina, it elicits an innate stabilizing reflex, the optokinetic reflex (OKR). There is a relatively simple wiring from retinal movement detectors to the eye muscles that compensate for the detected movement, making it smaller. In control engineering terms, this is a simple feedback regulator. The problem is that there is a significant delay between the compensatory control signals to eye muscles and sensory feedback from the signals. In control engineering this is called deadtime because from the viewpoint of the controller, the system appears to be unresponsive to any control signals.

Deadtime is considered to be one of the most difficult problems in control engineering. Consider how a feedback controller behaves in the face of deadtime: if the gain is high, the system starts oscillating wildly as a result of overcompensation that occurs during deadtime. If, on the other hand, the gain is low, the system is sluggish and doesn't properly compensate for errors—visual motion in this case.

After issuing a motor command, the controller should realize that a particular visual error has already been compensated for and that further compensatory movements are no longer necessary. This is clearly a task where a predictor might prove useful. There are at least two ways in which a predictor can be used:

Smith predictor: Predict the future state of the relevant sensory variable (such as visual motion in gaze stabilization) given a history of any relevant sensory and motor signals. Then use this predicted sensory variable as input to a feedback controller instead of the currently available (outdated) sensory input.

Feedback-error learning: Try to compensate for any errors before they are generated. Use the responses of a feedback controller as an error signal (not the target) for updating the motor predictions.

Roughly speaking, these two strategies are about sensory and motor prediction, respectively. The first strategy (alone) is called a Smith predictor after its inventor and the second one feedback-error learning because a feedback controller provides the error signal. These strategies can also be combined because sensory predictions can be useful inputs when making motor predictions.

The Smith predictor dates back to 1957 and is well-known and much studied in control theory. It can be quite sensitive to prediction errors which are bound to occur in real-world situations. In contrast, feedback-error learning seems to be more robust and a combination of the two where sensory prediction is used as an auxiliary variable for feedback-error learning should be able to combine the best of both worlds. In this paper I will concentrate on feedback-error learning.

In feedback-error learning, the predictor is supposed to predict and issue motor commands that silence the feedback controller. An obvious question that arises is how much in advance should the predictor try to issue the commands. Ideally this time would match the deadtime but it is usually difficult to know how long it is. If deadtime would be caused only by internal processing delays, it would at least be constant but unfortunately a significant portion of deadtime is often caused by inertia and other such properties of external world. These properties change from one task to another and correspondingly deadtime varies as well.

A simple solution is to spread the error signal temporally such that appropriate corrections are made at the optimal time in addition to corrections that are made at suboptimal times. On the one hand this solution could be considered as a feasible engineering solution: it will be more probable that the correct action will be taken at the correct time and in many cases it is more important to get the total amount of corrective movements right irrespective of their exact timing. On the other hand, it would be interesing to know how accurate timing the system can then learn? One might expect that the achievable accuracy depends on the amount of temporal spreading of the error signal. Rather surprisingly, this is not the case as will be demonstrated next.

Consider the following problem. There are 10 input channels $\mathbf{x}(t) = [x_1(t) \dots x_{10}(t)]^T$ which each carry information about the time. The n^{th} channel $x_n(t)$ in maximally active at time $t = n$. There is some overlap in the activations, in other words, the activations are spread temporally. In Figure 3, 10 input channels are active at times $1 \leq t \leq 10$ and are silent at times $11 \leq t \leq 20$. The task is now to find a linear mapping $y(t) = \mathbf{w}^T\mathbf{x}(t)$ of the inputs which outputs $y(t) = z(t)$ where the desired output $z(5) = 1$ and $z(t) = 0$ for $t \neq 5$.

If the error could be observed instantaneously $e(t) = z(t) - y(t)$, the problem could solved with standard supervised learning:

$$\Delta w \propto e(t)\mathbf{x}(t) \, . \qquad (1)$$

In this example the deadtime is five time steps: $e(t) = z(t-5) - y(t-5)$. This is the signal that we receive from the teacher. If we knew the delay, we could simply match $e(t)$ with $\mathbf{x}(t-5)$ in (1), but now we assume that deadtime is unknown for the learning system and the error signal $e(t)$ will be matched with several $\mathbf{x}(t-\tau)$ with various different τ according to

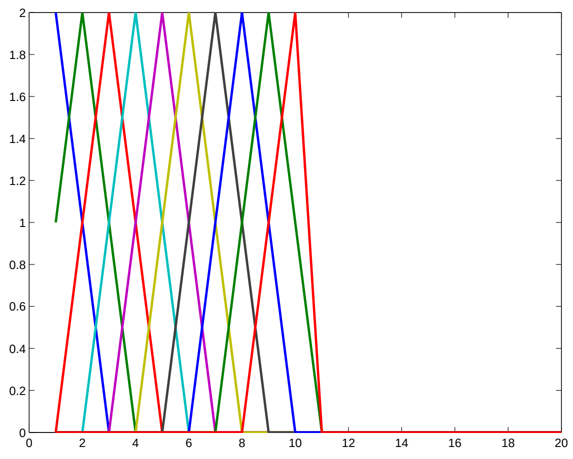

Figure 3: Ten input channels carry slightly overlapping information about the time.

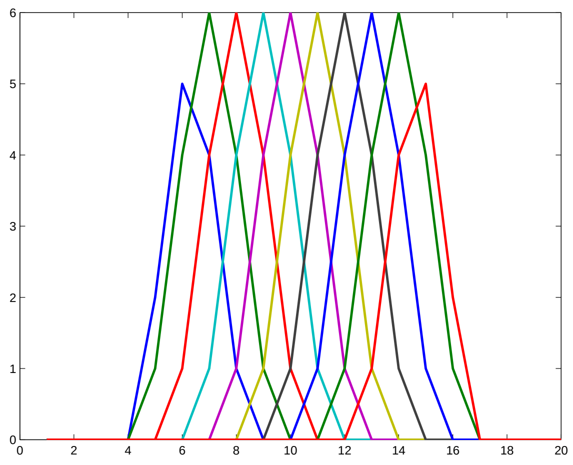

Figure 4: Eligibility traces which are achieved by delaying and temporally spreading the ten input channels in Figure 3.

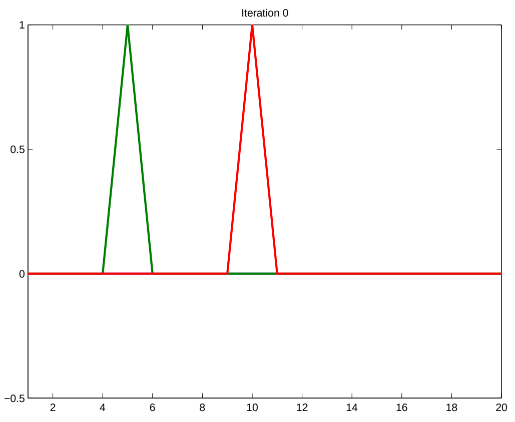

Figure 5: The weights are initialized to zeros and therefore the output (blue line behind other curves and doesn't show) is zero. The real error (green) occurs at time $t = 5$ but is observed (red) at time $t = 10$.

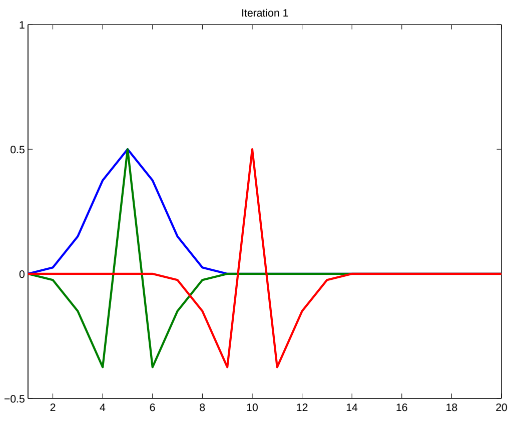

Figure 6: After the first weight update, the system responds (blue) around the correct time but temporal accuracy is low.

weighting kernel $h(\tau)$:

$$\Delta w \propto e(t) \int h(\tau)\mathbf{x}(t-\tau)d\tau \,. \qquad (2)$$

Note that in on-line learning system the matching needs to be implemented by delaying $\mathbf{x}(t)$ but conceptually this is equivalent to taking the error signal backward in time. The delayed inputs which are used in update rules are called eligibility traces. Figure 4 shows the eligibility traces of $\mathbf{x}(t)$ using delays of 4, 5 and 6 steps in proportions 1:2:1.

Figures 5–8 show how the response improves. The correct total amount of response is learned very quickly. Initially the response has poor temporal resolution which is expected because both the inputs and the error signal have poor temporal resolution. What is surprising is that after a large number of iterations the system converges to a response which is close to optimal.

In the above example the error signal was temporally centered at the correct delay and one might thus wonder whether convergence to optimal solution was simply due to the fortunate guess. This is not the case as shown by Figures 9–11. Although convergence is much slower, the iterations are clearly taking the system towards the correct solution. If 100,000 iterations sound like a lot, consider that people make about three saccades per second. Practice makes perfect.

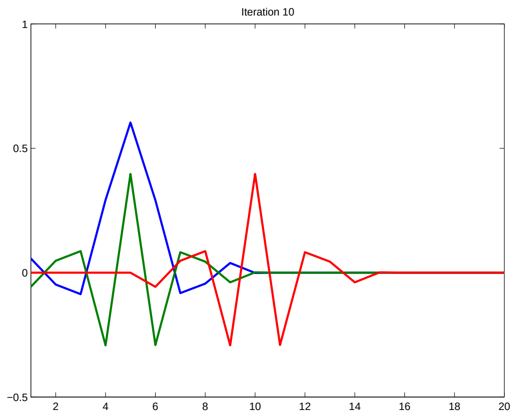

Figure 7: After 10 weight updates, the response has sharpened a bit.

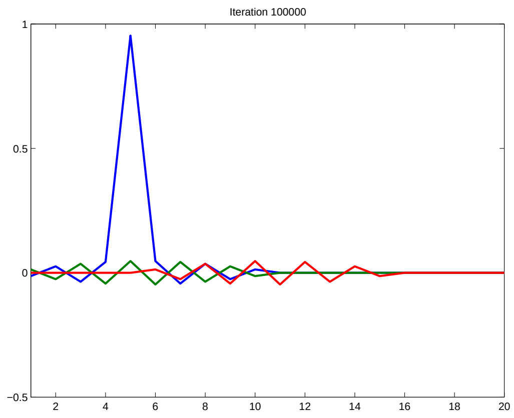

Figure 8: After 100,000 weight updates, the response is near optimal and certainly more accurate temporally than either the inputs or the error signal (after the spread).

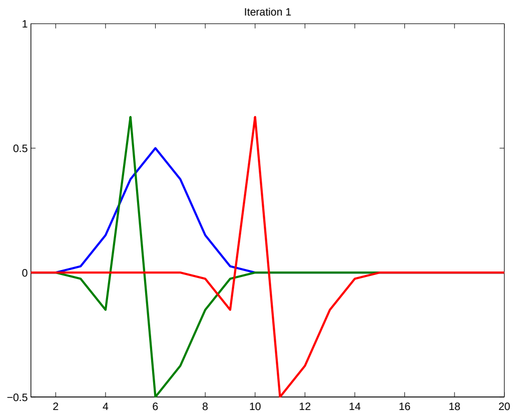

Figure 9: Delay is underestimated. After the first weight update, the system responds with low temporal accuracy around time $t = 6$.

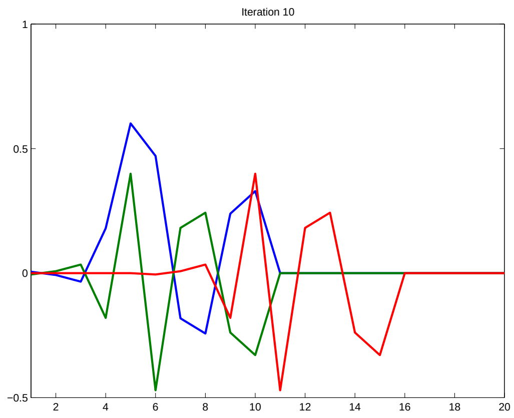

Figure 10: After 10 weight updates, the response has sharpened a bit and moved towards the correct delay.

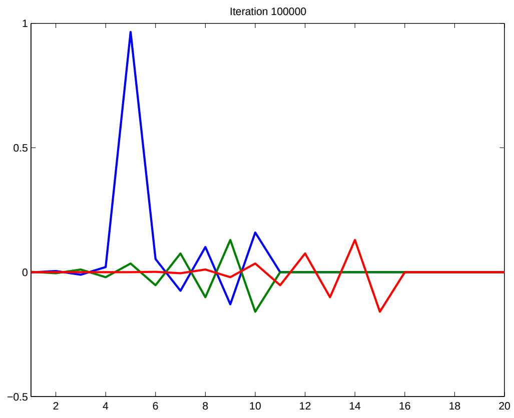

Figure 11: After 100,000 weight updates, the response is remarkably good although some amount of transient high-fequency ripple in the response is evident after $t > 5$. After around 300,000 iterations even that is almost gone.

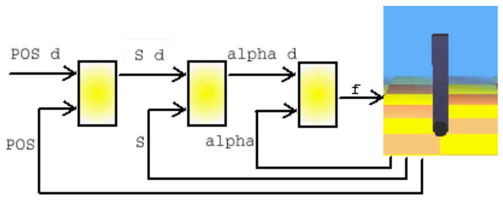

Figure 12: The "reflex" which provides the error signal for the predictive controller.

5 Robot simulations

I will next review experiments which demonstrate feedback-error learning in a simple but demanding robot control task. These simulations are taken from Heikki Joensuu's master's thesis (2006)[1] but similar work has been published for instance by Smith (1998) and McKinstry et al. (2006).

The robot is shaped like a pole and it moves in an arena by controlling its two wheels. In these simulations both wheels always had the same speed. Even though the robot cannot fall sideways, the body is quite unstable—quite like a unicycle. The robot was simulated using Webots[2] robot simulation environment.

5.1 Reflex

In feedback-error learning, the task of the controller is given in terms of a "reflex" or a feedback controller which gives a corrective motor signal, the error $e(t)$ in (2), if some sensory input level is non-optimal. In this simulation, the task of the robot is to stay in the middle of the arena in upright position. This is expressed in terms of a simple hierarchical P-controller (Figure 12) where each P-controller compares its desired value of a variable (reference value in control engineering terms) with the observed value and issues a control signal which is directly proportional to the difference (P stands for proportional). At the highest level is a position controller which outputs a desired speed in proportion to displacement from the center of the arena. The speed controller outputs a desired tilt angle for the robot: if the speed is optimal, stay upright; if the speed is too low, lean forward; and so on. The last controller tries to achieve the desired tilt angle by controlling the acceleration of the wheels.

Note that in this simulation, there were practically no internal processing delays. Due to inertia of the body, the controller will nevertheless need to anticipate. Although there is strictly speaking no deadtime, inertia works in much the same way and internal processing delays can therefore be handled by the same system. Also note that if the body changes—grows, for example—inertia changes and hence the optimal target advance in prediction changes, too.

[1]Videos and other material related to the thesis are available at `http://www.lce.hut.fi/research/eas/compneuro/projects/cerebellum/`.

[2]`http://www.cyberbotics.com/`

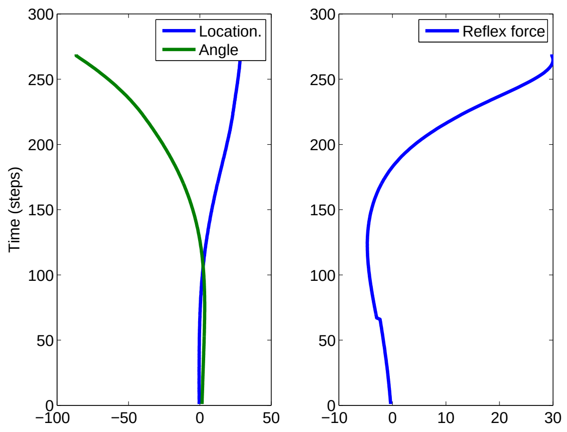

Figure 13: Behaviour with the reflex alone.

5.2 Results

As expected, the reflex alone cannot keep the robot up (Figure 13) but it can give useful corrections which can be used by a predictive controller to update its predictions. The target anticipation was on average 40 time steps and it was temporally spread. After 160 trials (Figure 14), the robot is able balance itself but does not succeed in centering itself (this is very typical behaviour during learning). After 300 trials (Figure 15), the robot can both balance and center itself rapidly and accurately.

In the above simulations, the input for the cerebellar model (the linear mapping described in Section 4) was "proprioceptive", concerning only its own body. A convenient property of predictors is that they can take almost any type of inputs; any information that will turn out useful will be used for making the predictions. Figure 16 depicts the behaviour of the robot after it has learned to anticipate a ball hitting it. The only change that needed to be made to the system was to include two input channels, one for balls approaching from the left and another for balls from the right. With more sensory inputs the controller could even learn to deal with balls of different properties.

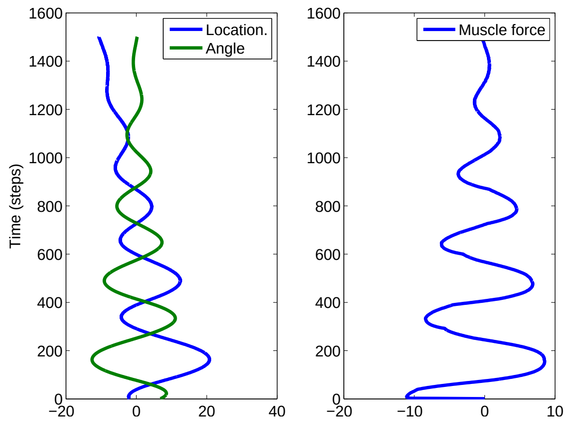

Figure 14: Behaviour after 160 trials.

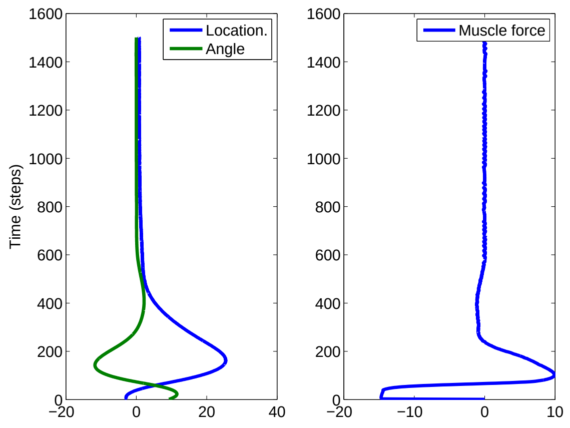

Figure 15: Behaviour after 300 trials.

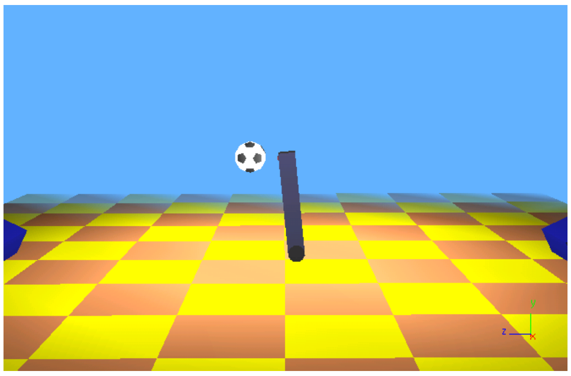

Figure 16: The robot has learned to make anticipatory movements before the ball hits it and is therefore able to cope with the impacts.

6 Discussion

The simulations discussed in this paper had just one degree of freedom while our own cerebellum controls hundreds of muscles. An interesting question is how this control generalizes to multiple degrees of freedom. I think that a feasible strategy for learning coordinated movements is to have independent reflexes for each degree of freedom (or reflexes that each control at most a few degrees of freedom) and to let the cerebellar model learn from experience how the movements in one part of the body need to be compensated by another part. This prediction task will be easier if the predictor for each degree of freedom gets information about the ongoing control signals sent to all other degrees of freedom.

Since the role of cerebellum in motor control is well understood, we can ask what should other parts of the brain do to help cerebellum achieve its job. One thing that cerebellum clearly needs is information which is useful as input for the predictor. For instance, if the mass distribution of the pole changes, the optimal control changes. It would therefore be useful to have system that monitors the dynamic properties of the body. In mammals, neocortex provides information about the state of the world, including the state of ones own body and its changing dynamic properties, intentions and goals, all of which are useful inputs for cerebellum to accomplish its control tasks.

One thing that the cerebellar system clearly cannot do is acquire new goals and reflexes. Not all goals can be predefined in terms of hard-wired reflexes and it is therefore necessary to have systems that learn new reflexes or otherwise generate corrective movements that lead to rewarding consequences. In mammals, basal ganglia and prefrontal cortex are known to collaborate in the generation of such goal-directed behaviour.

In conclusion, the "cerebellar algorithm"—prediction—is well understood on a general level. There are many (mutually compatible) ways in which a predictor can be used in control. In this paper, I focused on feedback-error learning. I showed that the temporal accuracy that can be achieved in such a strategy is surprisingly good, better than the temporal accuracy of the error signal. A used a simulated robot balancing task to demonstrate that feedback-error learning is, indeed, able to learn to control the unstable robot far better than the reflex that generates the error signals. The algorithm is computationally efficient and it is easy to use it as a module in a larger system.

Acknowledgements

I would like to thank M.Sc. Heikki Joensuu, who implemented self-supervised learning and the controllers for the robot in Matlab, and Mr. Harm Aarts, who developed the interface between Webots robot simulator and Matlab.

References

J. S. Albus. A theory of cerebellar function. *Mathematical Biosciences*, 10:26–61, 1971.

G. Allen, R. B. Buxton, E. C. Wong, and E. Courchesne. Attentional activation of the cerebellum independent of motor involvement. *Science*, 275:1940–1943, 1997.

R. Apps and M. Garwicz. Anatomical and physiological foundations of cerebellar information processing. *Nature Reviews Neuroscience*, 6:297–311, 2005.

A. G. Barto, A. H. Fagg, N. Sitkoff, and J. C. Houk. A cerebellar model of timing and prediction in the control of reaching. *Neural Computation*, 11:565–594, 1999.

H. Daniel and F. Crepel. Cerebellum: Neural plasticity. In M. A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*, pages 196–200. The MIT Press, 2nd edition, 2003.

M. Garwicz. Spinal reflexes provide motor error signals to cerebellar modules — relevance for motor coordination. *Brain Research Reviews*, 40(1):152–165, 2002.

J. S. Grethe and R. F. Thompson. Cerebellum and conditioning. In M. A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*, pages 187–190. The MIT Press, 2nd edition, 2003.

C. Hansel, D. J. Linden, and E. D'Angelo. Beyond parallel fiber ltd: the diversity of synaptic and non-synaptic plasticity in the cerebellum. *Nature Neuroscience*, 4:467–475, 2001.

C. Hofstötter, M. Mintz, and P. F. M. J. Verschure. The cerebellum in action: a simulation and robotics study. *European Journal of Neuroscience*, 16(7):1361–1376, 2002.

M. Ito. Historical review of the significance of the cerebellum and the role of purkinje cells in motor learning. *Annals of the New York Academy of Sciences*, 978:273–288, 2002.

H. Joensuu. Adaptive control inspired by the cerebellar system. Master's thesis, Helsinki University of Technology, Finland, 2006.

M. Kawato. Cerebellum and motor control. In M. A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*, pages 190–195. The MIT Press, 2nd edition, 2003.

M. Kawato and H. Gomi. The cerebellum and VOR/OKR learning models. *Trends in Neuroscience*, 15(11):445–453, 1992.

D. Marr. A theory of cerebellar cortex. *Journal of Physiology (London)*, 202:437–470, 1969.

J. L. McKinstry, G. M. Edelman, and J. L. Krichmar. A cerebellar model for predictive motor control tested in a brain-based device. *Proc Natl Acad Sci USA*, 103:3387–3392, 2006.

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*, 20 (14):5516–5525, 2000a.

J. F. Medina and M. D. Mauk. Computer simulation of cerebellar information processing. *Nature Neuroscience*, 3:1205–1211, 2000.

J. F. Medina, W. L. Noresa, T. Ohyama, and M. D. Mauk. Mechanisms of cerebellar learning suggested by eyelid conditioning. *Current Opinion in Neurobiology*, 10(6):717–724, 2000b.

T. Ohyama, W. L. Nores, M. Murphy, and M. D. Mauk. What the cerebellum computes. *Trends in Neurosciences*, 26(4):222–227, 2003.

S. P. Perrett, B. P. Ruiz, and M. D. Mauk. Cerebellar cortex lesions disrupt learning-dependent timing of conditioned eyelid responses. *Journal of Neuroscience*, 13:1708–1718, 1993.

R. L. Smith. *Intelligent Motion Control with an Artificial Cerebellum*. PhD thesis, University of Auckland, New Zealand, 1998. URL `http://www.q12.org/phd.html`.