

# Utilising a Cerebellar Model for Mobile Robot Control in a Delayed Sensory Environment

David Collins

Gordon Wyeth

Department of Computer Science and Electrical Engineering

University of Queensland

Brisbane, QLD Australia 4072

{collins|wyeth}@csee.uq.edu.au

## Abstract

Fast and accurate movement control for a system exhibiting significant feedback delay is traditionally a difficult problem to solve. In biological systems, it is thought that a part of the brain called the cerebellum overcomes such difficulties. This paper outlines the use of a cerebellar model in the control of a simulated mobile robot. The model is based around Albus's CMAC neural network, and uses the response of a non-delayed teaching module as a basis for learning. The model was able to produce results comparable to the teacher despite being subjected to severe sensory latency. After limited initial training the system can rapidly adapt to new situations and proved to have good generalisation between similar movements. The velocity profiles exhibited during learning were found to be similar to those of an infant.

## 1. Introduction

Using traditional control techniques, applications that require fast and accurate movements are typically limited by the latency associated with sensory data acquisition. To successfully execute such movements the controller needs immediate state information to reliably ascertain the correct control signal for the next portion of the movement. If the sensor data is unavoidably delayed, the control loop will suffer with the effects of using old data as the basis of the next motor command. This situation usually presents the undesirable options of carrying out a fast but oscillatory trajectory or an accurate but slow response, depending on the system gains. Neither of the two alternatives will meet the necessary requirements of both speed and accuracy. This scenario is all too common in many modern robotic applications, particularly with respect to machine vision, with frame latency introducing a large and mostly

insurmountable amount of uncertainty into the control loop. It is possible to improve the performance of such a system through the use of a model predictive controller. This technique often requires an accurate estimation of the system model, which in highly non-linear systems is difficult to ascertain. Advances in both sensor and computational technology are reducing the time between data sampling and usability, but as the need to make movements faster and more accurate increases, techniques that overcome the problem will be invaluable. With this in mind we turn to nature and see what adaptive mechanisms are exploited to solve this problem.

The human body as a system exhibits some striking similarities to the one described above. We have the ability to perform a stunning array of swift and precise movements, which for most of us occurs without so much as a conscious thought. But the attribute that makes our feats truly remarkable is the fact that nature has engrained in us some feedback latencies that theoretically should prevent successful execution of even the most basic human movements. Latency times in the human body range from 100ms for proprioception to 200ms for visual feedback (Kawato, 1995). With delays of this order how can the human body exhibit such accurate yet rapid voluntary movement? The answer lies in a part of the brain called the cerebellum (Kawato, 1995).

The cerebellum is thought to be nature's mechanism to eliminate this apparent anomaly. Research suggests that the cerebellum learns to produce coordinated movements through exposure to previous attempted movements. In doing so it learns to implicitly predict actual state parameters when presented with obsolete state information. Learning occurs on a continual basis providing infinite adaptability throughout the life of the system. Models inspired by the cerebellum have eliminated the need for a complex mathematical model of the plant, instead learning system parameters with the aid of a crude teaching module (Fagg *et al*, 1997b). Other theories of cerebellar function

suggest that the cerebellum can build an inverse model of the controlled system, hence virtually eliminating the need for feedback (Kawato and Gomi, 1992).

Within the robotics domain it is clear that any such system would have many tangible benefits. This paper outlines a cerebellar modeling approach as applied to the mobile robot domain. The study is designed to explore the cerebellum's ability to overcome the effects of sensory latency. More specifically the study aims to:

1. Determine if a cerebellar module can perform as well in the delayed environment as its teaching module can in the non-delayed environment.
2. Explore the generalisation ability of the cerebellum between similar movements.
3. Examine the models ability to continually adapt to new and unexpected conditions.

## 2. System Architecture

The task chosen to explore the relative effectiveness of a cerebellar model is outlined in Figure 1.

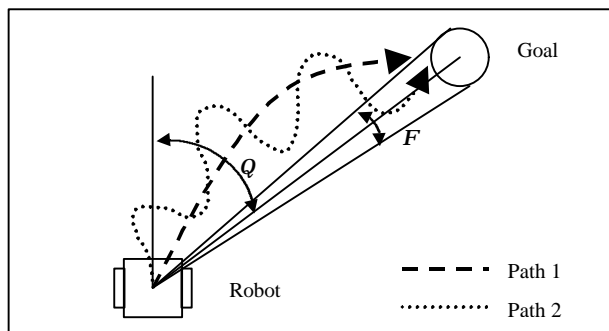


Figure 1: The Robot Task. Angles  $\Theta$  and  $\Phi$  represent the robots angular offset and width to the target respectively. Paths 1 and 2 show the TM's response in a non-delayed and delayed environment respectively.

The task consists of a mobile robot executing a goal approach trajectory. The simulated robot must approach the target quickly and smoothly, having time delayed sensor readings as its only source of input. The goal can be considered to be a spherical object or column. A movement is complete when the robot comes within a certain radius of the target. The sensors are designed to emulate vision with two quantities, *angular offset* ( $Q$ ) and *angular width* ( $F$ ) used to define the position and orientation of the robot with respect to the target. A simple control loop is used to link the sensors to each wheel motor. This hard-wired control loop reactively modulates the magnitude of the commands being sent to the motors depending on the current sensory inputs. Path 1 shows the trajectory of the robot if no sensory delay is present, where path 2 displays the oscillatory effects of introducing severe sensor latency into

the same simple control loop. This simple control loop, which is implemented via the *Teaching Module (TM)*, can only provide a smooth trajectory to the target if the system gains (output magnitudes) are reduced, resulting in a much slower movement. The TM, which as the name suggests is used as a reference behaviour for cerebellar learning, will be discussed in more detail later.

### 2.1 Modular Architecture

The modular architecture of the system is illustrated in Figure 2. The structure of the system is inspired by a model developed by Fagg *et al.* (Fagg *et al.*, 1997a). The Fagg model uses an array of *adjustable pattern generators* (APGs) (Berthier *et al.*, 1993; Barto *et al.*, 1995) to control a two-link arm in muscle space. A CMAC neural network was appended to the APG model to perform the function of a sparse, expansive *state encoder*. The APGs combine the current target position with delayed sensory and motor efference signals to produce motor commands that are intended to bring the arm to the specified goal. The system is trained using the commands from a crude corrective teacher, called the *extra-cerebellar* module, that assumes control of the arm if the cerebellar module fails to guide the arm to the desired target position.

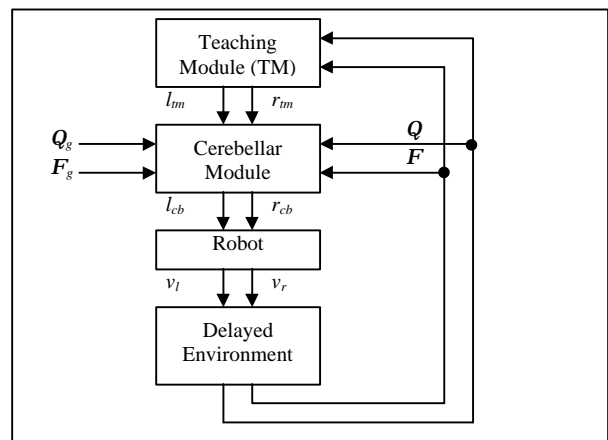


Figure 2: Modular Architecture.  $l_{tm}$  and  $r_{tm}$  represent the Teaching Modules left and right motor commands.  $l_{cb}$  and  $r_{cb}$  represent the same quantities for the cerebellar module.  $v_l$  and  $v_r$  are the actual motor velocities.  $\Theta_g$  and  $\Phi_g$  are the initial angular offset and width quantities of the goal, whereas  $\Phi$  and  $\Theta$  represent the current (delayed) state parameters.

Herein lies the main difference between the Fagg model and the model presented in this paper. In our model, the cerebellar module is always the source of descending motor commands. The teaching module is never allowed to control the robot directly, instead it provides suggested motor commands based on delayed sensory information. The cerebellar module uses these suggested commands for future learning, never for immediate action. In the Fagg model, the extra-cerebellar module actually gains control of

the arm to perform a corrective movement while the cerebellar module observes the resulting state changes to perform learning.

## 2.2 Robot Model

The simulated robot is based on a conventional differential drive system. The motor responses for each wheel are modeled as 1<sup>st</sup> order systems:

$$\frac{dv(t)}{dt} = \frac{k}{t} c(t) - \frac{v(t)}{t} \quad (1)$$

where  $v(t)$  is the wheel velocity,  $c(t)$  is the wheel command,  $k$  is a constant and  $t$  is the time constant.

## 2.3 The Teaching Module

The Teaching Module (TM) is based on a multisensorial Braitenberg vehicle (Braitenberg, 1984) as shown in Figure 3.

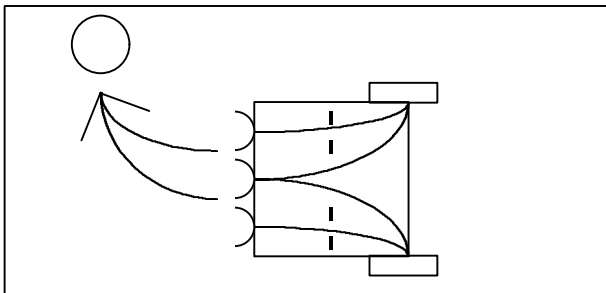


Figure 3: The Teaching Module (Braitenberg Vehicle).

Braitenberg vehicles of the style shown in Figure 3 can adopt a series of different behaviours. This is achieved by crossing the pathways between sensors and actuators and changing between excitatory and inhibitory synapses. This vehicle is coined a *love* vehicle because it always approaches the goal and will do so in a manner devoid of aggression.

The two lateral sensors in the model respond to the angular offset ( $Q$ ) of the robot. If the goal is sensed as being offset to one side of the robot, the command to the wheel on the goal side will be inhibited proportionally with the magnitude of the offset. If the offset angle is greater than 90 degrees, the wheel command is maximally inhibited causing it to stop. The other wheel will continue to move at the maximum base velocity causing the robot to turn sharply towards the goal. With the offset angle between zero and 90 degrees the level of inhibition varies between zero and 100 percent respectively. This results in both wheels travelling

at the maximum base velocity if the target is directly in front of the robot.

The centre sensor responds to the angular width ( $f$ ) parameter. It exhibits an inhibitory influence over both drive wheels equally. This sensor has no effect until the robot is close enough to the goal to trigger a threshold response ( $F = F_{threshold}$ ). When this occurs it will progressively bring the robot to rest over a few iterations, via a *stopping profile*. Figure 4 shows the responses for both the lateral and centre sensors.

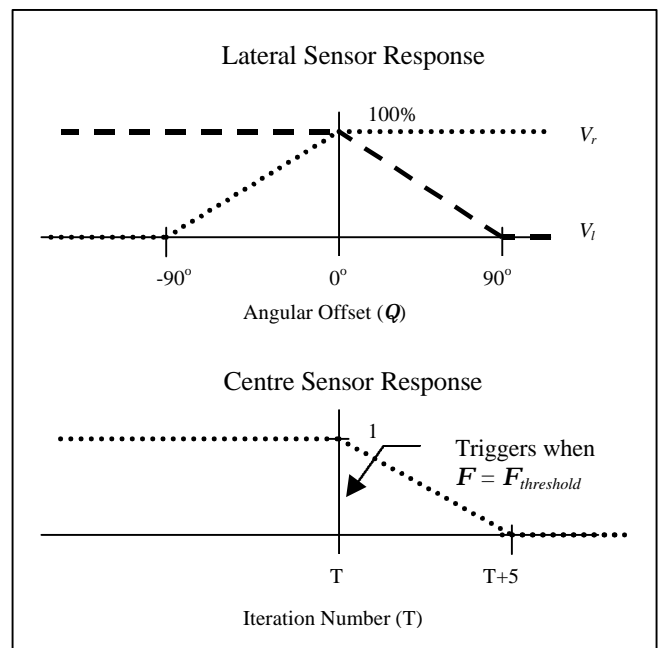


Figure 4: The Lateral and Centre Sensor Responses. The top graph shows the motor velocities  $v_l$  and  $v_r$  for different angular offsets ( $Q$ ). The bottom graph shows the *stopping profile*, which modulates each motor command once the angular width ( $F$ ) has reached the threshold ( $F = F_{threshold}$ ).

The simulator allows a fixed interval delay to be introduced between the sensors and the actuators. This leads to the situation where the motors could be trying to correct a left offset condition, based on the delayed data received, when the real time data indicates a right offset. The resulting movement will only serve to compound the problem leading to an undesirable oscillatory path.

It should be noted that the TM will not necessarily perfectly align the robot with the target, even with non-delayed input. The TM used was chosen for its simplicity and ability to produce smooth trajectories in the non-delayed environment and oscillatory trajectories in the delayed environment. The aim of the study is to see if the cerebellar module will replicate the non-delayed TM in the delayed domain, not perfectly align with the target.

## 2.4 The Cerebellar Module

The internal structure of the Cerebellar Module (CBM) is illustrated in Figure 5.

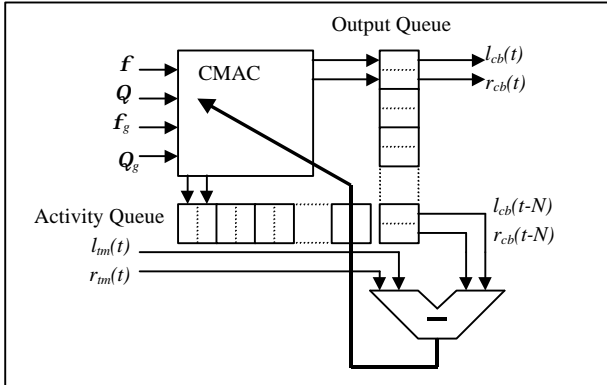


Figure 5: The Cerebellar Module. All quantities are the same as those outlined in Figure 2. The Output and Activity Queues store past cerebellar outputs and weight activity patterns respectively.

A CMAC (Cerebellar Model Articulation Controller) (Albus, 1975) neural network is the central feature of the cerebellar module. A CMAC exhibits the properties of an *expansive recoding network*, with inputs subjected to an expansive remapping in memory space such that similar input states share similar parts of memory, whereas significantly different input states are unlikely to share any

common memory. The signals then undergo a convergent mapping to produce an output. This results in a network that has good local generalisation properties, due mainly to an overlapping receptive field arrangement at the inputs to the network.

Figure 6 displays the structure of a simple four input, single output CMAC neural network (Miller *et al.*, 1992). Each input projects onto a set of input sensors with overlapping receptive fields (9 per input in figure 6). Each sensor turns ON if the input falls within its receptive field range. A group of adjacent sensors will be excited for each input. The number of sensors turned ON is equivalent to the receptive field width ( $f$ ) of each sensor ( $f = 4$  in figure 6).

The input sensors project to a series of *state space detectors* (SSDs). A sensor from each input is used as the input to each SSD. The SSD turns ON if all four input sensors are ON, hence performing a logical AND operation. The input sensors are methodically mapped to various SSDs in such a way as to ensure that the number of SSDs that are turned ON is equivalent to the field width parameter ( $f$ ) of the network. This mapping aids in the local generalisation ability of the network. If three of the four inputs remain fixed and the fourth is varied by a single quantisation band, then three of the four previously excited SSDs will remain ON.

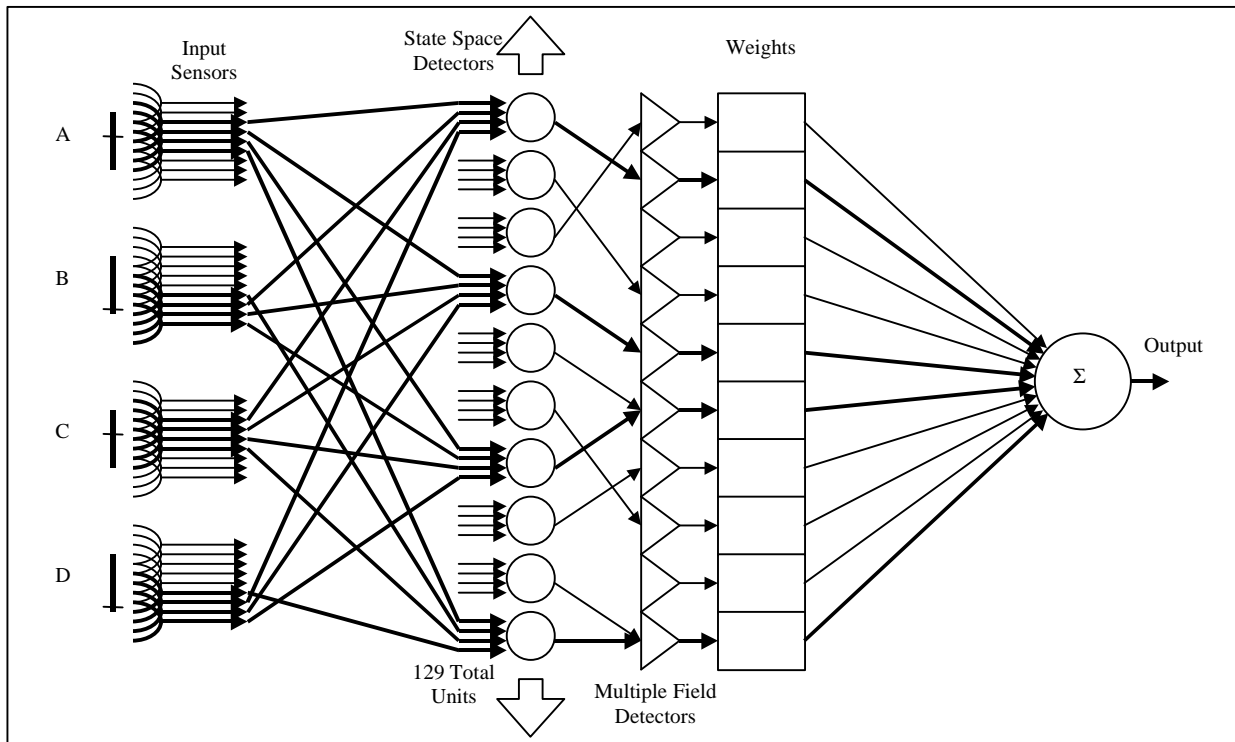


Figure 6: A Simple CMAC neural network with four inputs (A,B,C and D) and one output. Each input has 9 input sensors with a field width ( $f$ ) of 4. Note that only a partial set of the state space detectors has been shown (10 out of 129).

This means that the two similar input patterns will reference overlapping portions of the weight space in the network and therefore produce similar outputs. The network in figure 6 has a total of 129 SSDs. For clarity only 10 are shown and only the active SSDs have been connected to their respective input sensors.

Ideally there should be one adjustable weight attached to each SSD in the CMAC. In reality the potentially large number of SSDs in a network will make this impractical, necessitating the use of a smaller set of weights. To perform this convergent mapping a *multiple field detector* (MFD) is assigned to each weight. Several SSDs can map to the same MFD. This introduces a noise component to the network, with the possibility of two dissimilar input patterns invoking the same weight. The effect of this noise can usually be managed to within tolerable limits. Each MFD performs a logical OR function. In practice each SSD is assigned a virtual index which is then passed through a hashing function to select a weight. Each of the weights selected by an active MFD is summed to produce the network output.

The CMAC neural network was developed as a result of Albus's theory of cerebellar function (Albus, 1971). The cerebellar cortex receives information via *mossy fibres* and *climbing fibres*. Mossy fibres provide plant state and contextual information, whereas climbing fibres are thought to convey error information for learning. The input sensors in the CMAC are analogous to the *granule cells* in the cerebellum which are excited by the mossy fibres. The connections projecting from the input sensors represent *parallel fibres*. These parallel fibres eventually converge to a single *purkinje cell* (the summing node in figure 6) via a set of modifiable synapses.

In the cerebellar module, the CMAC receives as input (via mossy fibres), the current (delayed) and initial angular widths and offsets, with the initial quantities held constant throughout the time course of the movement.

The CMAC produces left and right wheel commands as outputs which are passed through to both the motors and the output queue. The output queue stores past CMAC responses that will eventually be compared to the TM response during future learning. Biologically this error signal would be sent back to the cerebellum via the climbing fibres.

Any connections in the network that are responsible for producing the current output have their indices recorded and stored in the activity queue. This structure acts as a register of past cerebellar activity, again to assist in future learning.

### 3. Learning

For the system to learn to perform as well as a non-delayed TM in the delayed environment and be adaptive to new

situations, it must address the issues of structural and temporal credit assignment.

Structural credit assignment identifies which connections in the network contributed to the production of an undesirable response. Tagging the weights that are influenced by these connections enables the learning process to specifically modify only the weights that were involved in this response and not the weights that were dormant at the time and probably aligned with a different class of movement.

Temporal credit assignment is used to record the history of structural credit assignment. This is particularly important in the application under consideration, as a record of connection activity is required later to enable effective delayed learning. The activity queue is used to store a history of active synapses as a function of time. This is analogous to an *eligibility trace* (Sutton and Barto, 1990).

By introducing a delay in the feedback pathway of the system, the current state parameters  $\mathbf{f}(t)$  and  $\mathbf{Q}(t)$  reach the controller in the form  $\mathbf{f}(t-N)$  and  $\mathbf{Q}(t-N)$ , where  $N$  is the delay parameter in iterations. The CMAC will respond with the commands  $l_{cb}(t)$  and  $r_{cb}(t)$ , but will have the past commands,  $l_{cb}(t-N)$  and  $r_{cb}(t-N)$ , stored in the output queue. These variables correspond to the cerebellar response  $N$  iterations previously, when the supplied data was current. The TM produces the desired commands for the delayed state information,  $l_{tm}(t)$  and  $r_{tm}(t)$ , which when compared with the retrieved cerebellar commands produce an error signal for supervised learning. The weights ( $\mathbf{W}(t-N)$ ) active  $N$  iterations earlier are fetched from the activity queue and modified during learning. This scheme should encourage the CMAC to predict the TM response to the current state parameters given only delayed information. The learning algorithm is in the form of the least mean square (LMS) rule:

$$\Delta w_i(t-N) = \frac{\mathbf{a}}{f} (c_{tm}(t) - c_{cb}(t-N)) \quad (2)$$

where  $w_i(t-N)$  is the active weight selected from the activity queue,  $f$  is the field width parameter for the CMAC inputs,  $\mathbf{a}$  is the learning rate parameter,  $c_{tm}(t)$  is the current teaching module motor command and  $c_{cb}(t-N)$  is the cerebellar command selected from the output queue.

This paradigm requires learning to be started only when the queues have been filled and should continue after the movement has terminated, to empty the queues. This *learn past termination* ensures that the motion performed close to the goal will be learnt.

## 4. Results

### 4.1 Delayed Teaching Module Behaviour

The first experiment examines the effects of varying sensory delay to the teaching module with no cerebellar involvement. The robot is placed at the position corresponding to (300cm, 100cm) in Cartesian co-ordinates, with its heading aligned with the negative x-axis. The target is placed at (300cm, 300cm), 200cm to the right of the robot, corresponding to an angular offset of  $90^\circ$  and an angular width of  $8.58^\circ$ . The stopping profile threshold ( $F_{threshold}$ ) is set to 20 degrees. Figure 7 shows trajectory plots for 4 different delay parameters. With a data sampling period of 100ms, the delay parameter is varied between 100ms (sampling delay), 300ms, 500ms and 700ms.

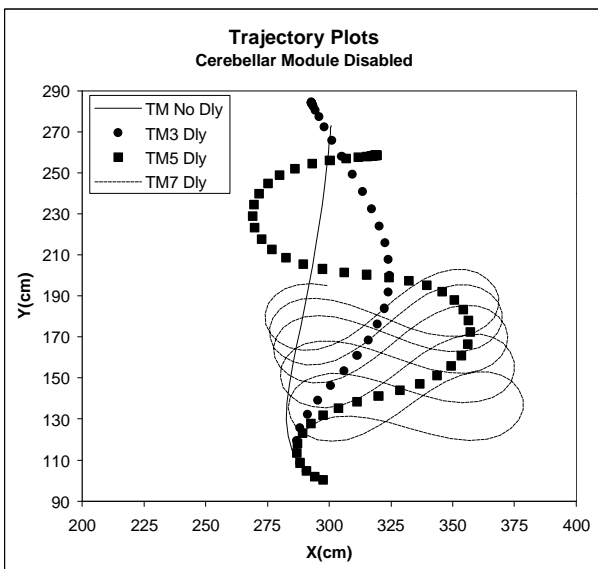


Figure 7: X-Y Trajectory Plots of TM responses with varying delay. The paths show TM responses for delays of 100ms (No Delay), 300ms, 500ms and 700ms respectively. The cerebellar module has been disabled.

It can be seen that the relative amount of deviation that the robot experiences on the approach varies proportionally to the amount of delay. The non-delayed TM executes a direct path to the goal whereas the TM delayed by 700ms will not be able to get to the goal as it gets trapped in a cyclical “figure-of-eight” loop.

### 4.2 Single Point Trajectory

The second experiment conducted on the system consists of repeated trials from the same starting point, to demonstrate the emerging learning pattern of the cerebellar module. The CMAC neural network is configured with 65 receptive fields for each input quantity. The network has a receptive field width ( $f$ ) of 5, meaning 5 of the 65 input units will be excited for any given input value. This parameter relates to the local generalisation capability of the network, the larger the field width with respect to the field number, the greater the influence a given input state has on surrounding input states. Each input can be separated into 61 discrete states resulting in  $(61)^4$  possible state combinations being separable into discrete state partitions. The learning parameter ( $\alpha$ ) is set to 0.5 and all the CMAC weights are initially set to zero. For all the experiments herein, the data sampling period is 100 ms and the sensory delay parameter is set to 700 ms. The stopping profile threshold ( $F_{threshold}$ ) is set to 20 degrees.

In the second experiment the robot is again placed at the position corresponding to (300cm, 100cm) in Cartesian co-ordinates, with its heading aligned with the negative x-axis. The target is placed at (300cm, 300cm), 200cm to the right of the robot, corresponding to an angular offset of  $90^\circ$  and an angular width of  $8.58^\circ$ . The robot is then made to execute a series of trials each replicating the same initial conditions described above. Figure 8 displays the evolving competence of the cerebellar module for trials 1, 3, 10 and 30 respectively. The average wheel velocity, otherwise known as the translational component of the motion, is plotted against time for both the cerebellar response in the delayed environment and the TM response in the non-delayed environment (desired response).

It can be seen that by trial 30, the cerebellum has learnt to closely replicate the response of the non-delayed TM, finishing with a relatively smooth motion in approximately 2.8 seconds. In trial 1 the cerebellar module can only manage to complete the movement in 9.7 seconds, utilising 3 distinct velocity peaks in the process.

Figure 9 displays the x-y trajectory plots of the experiment. It should be noted that the x-axis in Figure 9 has been expanded against the y-axis to separate the trajectory points in a 5 to 1 ratio. The pulsing velocities exhibited on trial 1 are evident by the clustering and expansion of trajectory points, each representing a 100 ms sample

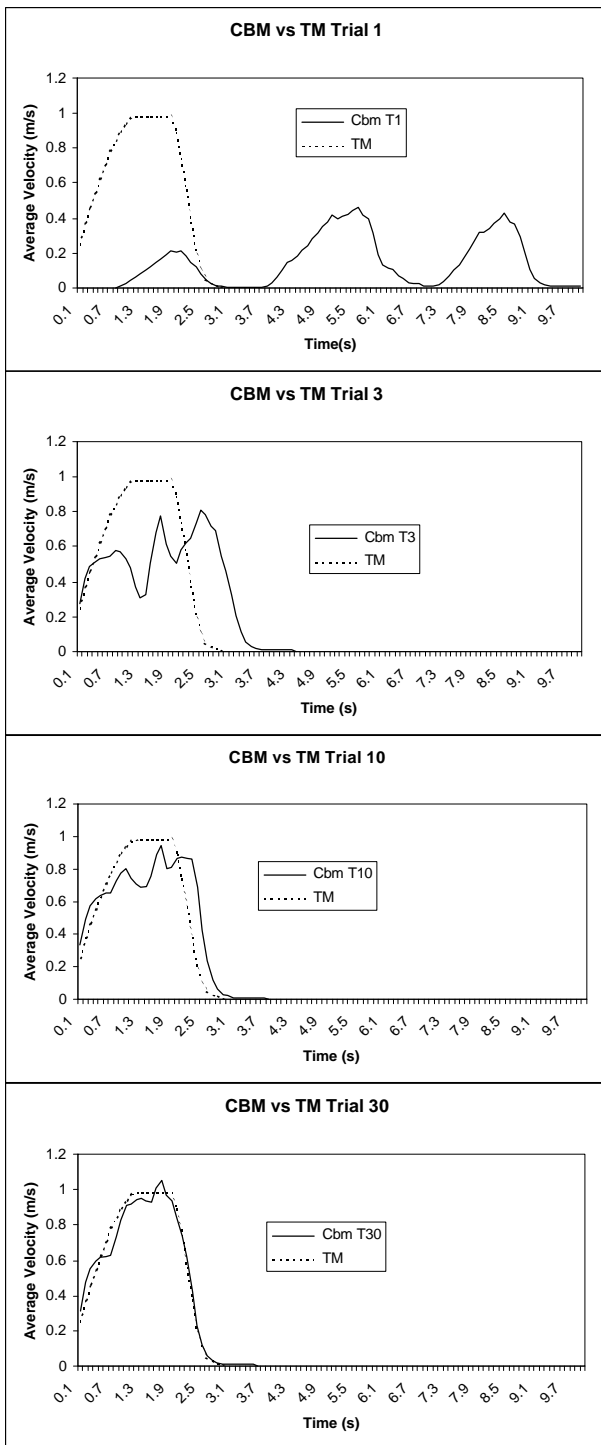


Figure 8: Cerebellar Learning. CBM velocity response in the delayed environment versus the TM response in the non-delayed environment (desired response), for trials 1, 3, 10 and 30 respectively.

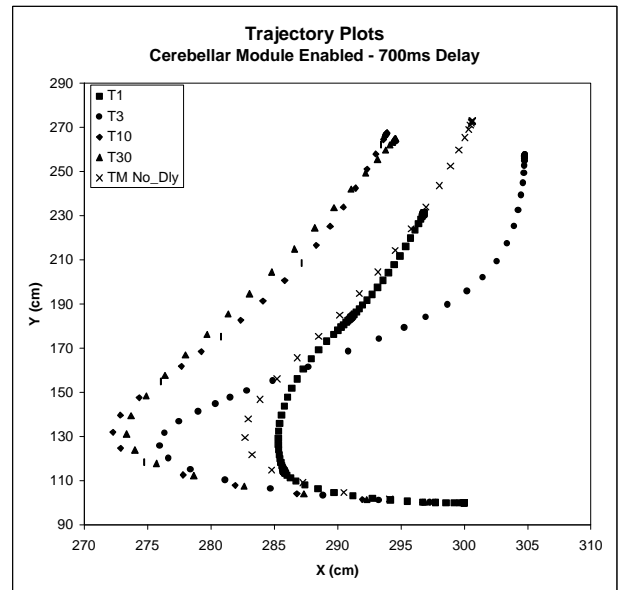


Figure 9: X-Y Trajectory Plots. Paths executed for trials 1,3,10 and 30, as well as the course executed by a non-delayed TM. Note the expanded x-axis with respect to the y-axis.

### 4.3 Multiple Point Trajectory Generalisation

To examine the generalisation capabilities of the network, 150 random starting positions were generated for both a training and testing set. The random co-ordinates were limited to a square with borders 2 m above, below, left and right of the goal, with the region 0.5 m from the goal excluded. This area can be segmented into approximately 2100 discrete state partitions in the CMAC. With 150 random starting positions, this equates to the system being exposed to 7% of all possible starting locations. In practice this figure may be less than 7% as the randomly generated training set may map a number of starting locations to the same discrete state partition. The robot heading was held aligned with the negative x-axis, with the position of the points with respect to the target producing angular widths and offsets of varying degrees. The network and delay parameters were the same as those selected for the second experiment.

The robot executed the first training set 10 times. In the first epoch, the average time of an approach was 4.56 seconds and the robots average displacement at the end of the movement from the goal centre was 58.2 cm. These figures improved to 2.26 seconds and 39.86 cm respectively by epoch 10. The non-delayed TM produced 2.26 seconds and 31.50 cm for these quantities respectively. The average starting displacement between the centres of the robot and goal respectively was 154.48 cm.

The system then navigated the testing set once, recording 2.60 seconds for average execution time,

corresponding to a 15% increase on the training set, and 48.73 cm for goal centre displacement. Exposure to a second epoch of the testing set quickly reduced the quantities to 2.30 seconds, (only 1.7% higher than the training set) and 44.61 cm respectively. Table 1 displays these results.

	Average Execution Time (s)	Average Final Goal/Robot Centre Displacement (cm)
TM Non-Delayed	2.26	31.50
<b>Training Set</b>		
CBM Epoch 1	4.56	58.20
CBM Epoch 10	2.26	39.86
<b>Testing Set</b>		
CBM Epoch 1	2.60	48.73
CBM Epoch 2	2.30	44.61

Table 1 – Training and Testing set generalisation experiment results. Each set contains 150 random starting points for a goal approach trajectory.

## 5. Discussion

The first experiments showed the effect of introducing a delay between the sensors and the actuators. The ability of the robot to approach the goal was progressively impeded to the point where it could not longer progress towards the goal. This is evidenced in the 700ms delayed plot, with the robot converging to a “figure of eight” loop and staying there.

In the single point trajectory experiment the system achieved a good match between the velocity profiles of the delayed cerebellar module and the non-delayed TM. The fact that a similarly delayed TM response failed to complete the task due to a lack of convergence highlights the significance of this result.

Perhaps the most interesting characteristic of the trial was the velocity profiles produced by the cerebellum during learning. Studies suggest that when an infant first learns to execute a reaching movement, the limb exhibits several peaks in its velocity profile (von Hofsten, 1979). This is in direct contrast to a similar adult attempt, which displays a single peaked, bell shaped profile. It has been argued (Fagg *et al*, 1997b) that the multi-peaked infant profile is due to a corrective mechanism external to the cerebellum producing pulsing trajectories in the direction of the goal. The authors hypothesize that there may be a simpler explanation that

does not require an external mechanism to produce the pulsed behaviour.

With the CMAC weights initialised to zero, coupled with the fact that descending motor commands must come from the cerebellar module, the robot must first learn to respond to the current state parameters if it is to move.

When sufficient time has passed to fill the queues, a learning operation will occur causing the cerebellar module to produce a small non-zero output on the subsequent iteration. This causes the robot to move into a different state space region. Due to the relatively small magnitude of the velocity command, the new region should significantly overlap with the first, resulting in it sharing a portion of the weight increase. After further learning the velocity command is larger than the last, displacing the robot into a progressively dissimilar region. This continues until the robot ventures into a region that has minimal learning overlap with the previous region, resulting in a near-zero velocity command, causing the process to repeat. As a consequence of this behaviour, more peaks will be visible in the initial trial if the width of local generalisation ( $f$ ) is reduced. This is shown in figure 10 when field width is reduced to 3.

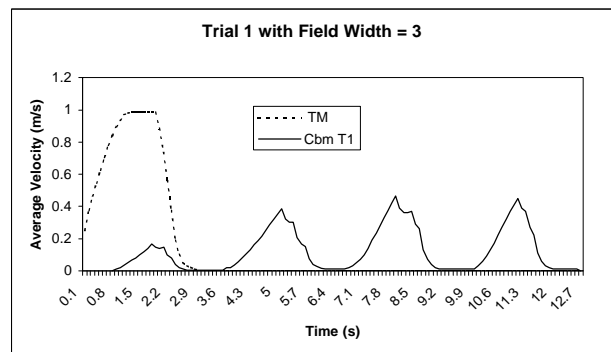


Figure 10: Reduced field width ( $f$ ) velocity response.

The current model does have the disadvantage of being unable to respond to changing conditions until the initial latency period has elapsed. The velocity profile of trial 30 shows an initial 1<sup>st</sup> order step response converging to 0.6  $\text{ms}^{-1}$  for the initial 700 ms of the trial. After this time, the cerebellar module attempts to track the desired response. Although the cerebellar module learns to respond to the initial state parameters, it cannot actively change its response in the initial latency period as the sensor data indicating movement has not yet reached the controller. This effectively means that the state parameters presented to the input of the CMAC network cannot change for the initial 700ms, hence the outputs of the network will remain fixed for this initial period as well. This behaviour is evident in the trajectory plot of trial 30. It indicates a slower rate of turn in the initial stages of the movement, resulting in a slight deviation from the desired trajectory. Despite this the system still manages to produce results comparable to the

reference trajectory and significantly better than the non-delayed TM.

The generalisation experiment showed that the system used previous exposure to similar states to adapt rapidly to new situations. With all the weights cleared, the first epoch of the training set took 4.56 seconds on average to complete a movement. After training was complete the first epoch of the testing set saw this figure reduce to 2.6 seconds. Although the cerebellar module had not been exposed to the testing set it was able to perform significantly better using the experience of similar movements learnt during training. The second epoch of the testing set brought the average execution time down to within 1.7% of the last training set presentation (epoch 10). This is an encouraging result given the relatively sparse distribution of the training set, which represented less than 7% of all possible starting locations (as distinguishable by the CMAC). This indicates that the local generalisation property of the CMAC assisted in providing some form of useful response for a previously unseen starting location. State information for a new starting location overlapped with previously learnt state information from the training set. This results in the sharing of common weights between the two input patterns that were modified during training for that class of movement. Hence, even for a previously unseen input pattern, the network will already contain partial knowledge of the correct response. This results in rapid convergence to the correct response and also allows the network to provide a proportionally correct response for the current motor command.

The system appears to have evolved into a predictive controller that uses delayed state information to implicitly predict the current state parameters and generate the action corresponding to the current state. To do this effectively the system must have made previous attempts of the movement. The response of the teaching module must be consistent for a given set of state parameters. An erratic or random reference trajectory will cause the prediction mechanism of the cerebellar module to fail. This results in the CMAC network associating incorrect responses with a given set of input conditions.

The model presented thus far will try to reproduce the TM response in a non-delayed environment. This response is a descending motor command. Whether or not the robot will respond with matching velocities is a property of the system. The system has no form of low level compensator like a traditional PID loop to provide regulation and ensure that the motor command is adhered to. In biological systems, once a descending motor command is issued, low level spinal reflexes perform a type of local servo loop, to improve system responses. The cerebellum simply modulates these commands at a higher level to implement motor control.

One possible modification to the system to achieve better response is to incorporate the delayed actual velocities into the learning system. This way the cerebellum would adapt such that the desired velocity would match the actual velocity. This is a subject for further investigation.

## 6. Conclusions and Future Work

A cerebellar control architecture for a mobile robot in a sensory delayed environment has been presented. The system demonstrated the ability to perform as well in a time delayed environment as the teaching module did in the non-delayed environment. The system also exhibited impressive generalisation and adaptation qualities, being capable of rapidly learning new situations after prior exposure to similar situations.

Some issues brought to light by this study will be the focus of immediate future work. Methods to overcome the initial delayed reaction of the cerebellar module will be examined, along with using the cerebellar velocity response in learning instead of the descending motor command.

Following these investigations the study will be introduced to the real domain of robot soccer. Instead of approaching a stationary target, the model will be adapted to strike a moving ball to a specified goal, using delayed feedback provided by a global vision system. It is expected that some reinforcement learning techniques will be appended to the current system to assist in developing a selection of correct actions for a given situation.

The ultimate goal is to develop a system that can assess how well it is performing (i.e. did the last shot at goal go in ?) and continually adapt in response to poor performance and changing internal and environmental conditions.

## References

- Albus J. S. (1971), "A Theory of Cerebellar Function," *Mathematical Biosciences*, 10, pp. 25-61.
- Albus J. S. (1975), "A New Approach to Manipulator Control: The Cerebellar Model Articulation Controller (CMAC)," *Transactions of the ASME, Journal of Dynamic Systems, Measurement and Control*, pp. 220-227.
- Barto A.G., Buckingham J.T. and Houk J.C., (1995), "A predictive switching model of cerebellar movement control," In D.S.Touretsky, M.C. Mozer and M.E. Hasselmo, editors, *Advances in Neural Information Processing Systems*, 8, pp. 138-144. MIT Press, Cambridge, Massachusetts.
- Berthier N.E., Singh S.P., Barto A.G. and Houk J.C.,(1993), "Distributed Representation of Limb Motor Programs in

- Arrays of Adjustable Pattern Generators," *Journal of Cognitive Neuroscience*, 5, pp. 56-78.
- Braitenberg V. (1984), *Vehicles: Experiments in Synthetic Psychology*, MIT Press.
- Fagg A. H., Sitkoff N., Barto A. G. and Houk J. C. (1997a), "Cerebellar Learning for Control of a Two-Link Arm in Muscle Space," *Proceedings of the IEEE Conference on Robotics and Automation*, pp. 2638-2644.
- Fagg A. H., Zelevinsky L., Barto A. G. and Houk J. C., (1997b) "Using Crude Corrective Movements to Learn Accurate Motor Programs for Reaching," *Extended Abstracts of the NIPS\*97 Workshop, Can Artificial Cerebellar Models Compete to Control Robots?*, Chapter 6, pp. 20-24.
- Kawato M., (1995) "Cerebellum and Motor Control," *The Handbook of Brain Theory and Neural Networks*, MIT Press, Cambridge, Massachusetts, pp. 172-178.
- Kawato M. and Gomi H., (1992) "A Computational Model of Four Regions of the Cerebellum Based on Feedback-Error Learning," *Biological Cybernetics*, 68, pp. 95-103.
- Miller W.T., Glanz F.H. and Kraft L.G., (1992), "CMAC: An Associative Neural Network Alternative to Backpropagation," In C.Lau, editor, *Neural Networks. Theoretical Foundations and Analysis*, pp. 233-240. IEEE Press, New York, NY.
- Sutton R. S. and Barto A. G., (1990) "Time-Derivative Models of Pavlovian Reinforcement," In M. Gabriel and J. Moore, editors, *Learning and Computational Neuroscience: Foundations of Adaptive Networks*, pp. 497-537. MIT Press.
- von Hofsten C., (1979) "Development of Visually Directed Reaching: The Approach Phase," *Journal of Human Movement Studies*, 5:160-168.