Cerebellar Joint Compensation for a Humanoid Robot

Damien Kee and Gordon Wyeth
School of Information Technology and Electrical Engineering
University of Queensland, Brisbane Australia 4072
{kee, wyeth}@itee.uq.edu.au

Abstract:
The loads seen at each joint of a humanoid robot during various points in a gait can fluctuate greatly, often by an order of magnitude. For example the load seen at the ankle during the swing phase is inconsequential compared to the load seen when the same ankle is in the single support phase of a typical walking gait. As a result, feedback control techniques perform poorly in minimising trajectory errors and cope poorly with unforeseen changes in system dynamics. This paper explores the addition of an adaptive control system inspired by the architecture of the cerebellum to improve system response. The cerebellar architecture learns to compensate the changes in load that occur during a cycle of motion. The joint compensation scheme, called Trajectory Error Learning, is used to augment the existing feedback control loop on a humanoid robot. The results from tests on the GuRoo platform show an improvement in system response for the system when augmented with the cerebellar compensator.

1 Introduction

Complex robots, with high degrees of freedom are becoming more common place in today's society. Robots with multiple limbs such as humanoids and octopeds are becoming more prominent in areas as varied as domestic robotics and all-terrain exploration. Such systems are difficult to model mathematically and hence analytical determination of feed forward dynamics for model based control can be both a complicated and time consuming process. In addition to this, these robots are progressively moving from a structured environment into regular society. Contact with the real world and human interaction further complicates the system loads.

Conversely, biological controllers do not use an accurate model of the system, rather incremental adjustment of the control parameters is performed, based on the experience of the system. Initial response may be quite crude, but over time appropriate control parameters are learnt. Changes in system dynamics such as an increase in size are accommodated by the continually updating controller.

Neural networks hold some promise in the field of trajectory control with the ability to 'learn' system dynamic without explicit representation of a robot's configuration. Sudden changes in system dynamics such as an additional payload, or an unexpected disturbance can be accounted for without the need to change control parameters.

This paper uses Trajectory Error Learning (TEL) [Collins, 2003], based on a CMAC neural network, to assist a conventional PI controller with trajectory tracking. The GuRoo humanoid robot with its high degree of freedom and non-linear dynamics forms a suitable platform to apply the system.

1.1 Previous Work

The use of a cerebellum models for motion control has been studied in the past. Infants of approximately 5 months of age display multiple accelerations and decelerations when moving an arm [Barto 1999]. This series of sub-movements eventually guides the arm to the desired position. Over time, and with more experience, the child learns the required muscle movements to smoothly guide the arm. This shows that the human body is not born with a perfect plant model, but in fact learns it through experience. As the child grows and the system dynamics change, the controller is updated to maintain suitable control responses.

Collins and Wyeth [2000] used a CMAC to generate the required velocities needed for mobile robot to move to a waypoint. Significant sensory delay was introduced that would cripple a traditional control system. The CMAC was able to learn the system dynamics, compensate for this delay and produce the required signals necessary to move to the waypoint. A smooth velocity profile to the goal was learnt from an initial response consisting of several short burst of motor activity.

Fagg *et al* [1997] implemented a CMAC control system on a 2 degree of freedom arm, actuated by three opposing sets of muscles. The CMAC is responsible for the co-ordination of these three actuators to control the two joints. When the CMAC does not bring the arm to the required position, an additional external CMAC was engaged that produces short sharp bursts of motor activity until the target was reached. Once the desired position was reached, the trial was terminated and a new trial initiated. Over time, the external CMAC was made redundant as the original CMAC correctly learned the required muscle commands.

1.2 Paper Overview

Section 2 describes GuRoo, the humanoid platform constructed at the University of Queensland, on which the research is applied. Section 3 outlines the CMAC neural network used as the basis for learning. Section 4 outlines the difficulty in using the current conventional control and described the application of Trajectory Error Learning (TEL). Section 5 describes the crouching experiment undertaken and presents results from before and after the implementation of the system. The final section draws conclusions from these results and discusses where these results may lead.

2 GuRoo Project

GuRoo is a fully autonomous humanoid robot (Figure 1) designed and built in the University of Queensland Robotics Laboratory [Kee *et al*, 2003]. The robot stands 1.2 m tall has a total mass of 34 kg, including on-board power and computation. *GuRoo* is currently capable of a number of demonstration tasks including balancing, walking, turning, crouching, shaking hands and waving.

The intended challenge task for the robot is to play a game of soccer with or against human players or other humanoid robots. GuRoo has been designed to mimic the human form and function to a degree, with considering conflicting factors of function, power, weight, cost and manufacturability.

Figure 1 also shows the degrees of freedom contained in each joint area of the robot. In the cases where there are multiple degrees of freedom (for example, the hip) the joints are implemented through short sequential links rather than as spherical joints.

2.1 Electro-Mechanical Design

The robot has 23 joints in total. The legs and spine contain 15 joints that are required to produce significant mechanical power, most generally with large torques and relatively low speeds. The other 8 joints drive the head and neck assembly, and the arms. The torque and speed requirements are significantly less. Table 1 outlines the type and axis of actuation of each motor. Due the high power / low velocity nature of these joints, large gearboxes are used which contribute to the length of the actuators and hence the unnaturally wide legs.

The centre of gravity of each leg lies outside the line of the hip rotation, and as such, the legs naturally swing inwards. The motors that drive the roll axis of the hip joints are each supplemented by a spring with a spring constant of 1 Nm/degree. These springs serve to counteract the natural tendency of the legs to collide, and help to generate the swaying motion that is critical to the success of the walking gait.

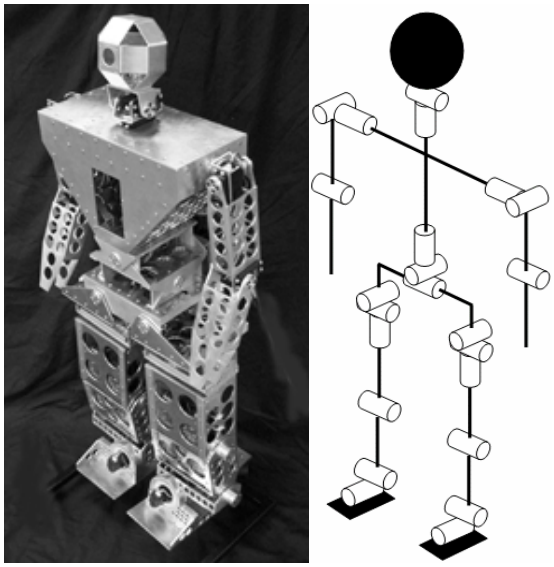

Figure 1: The GuRoo humanoid robot with a schematic showing the positions and order of joints.

Table 1: Type and axis of each DoF. "2 x" indicates a left and right side.

Joint	Type	Axis	No.
Head/Neck	RC Servo	Pitch + Yaw	2
Shoulder	RC Servo	Pitch + Roll	2x2
Elbow	RC Servo	Pitch	2x2
Spine	DC Brushed	Pitch + Roll + Yaw	3
Hip	DC Brushed	Pitch + Roll + Yaw	2x3
Knee	DC Brushed	Pitch	2x1
Ankle	DC Brushed	Pitch + Roll	2x2
		TOTAL	23

2.2 Distributed Control Network

A distributed control network controls the robot, with a central computing hub that sets the goals for the robot, processes the sensor information, and provides coordination targets for the joints. The joints have their own control processors that act in groups to maintain global stability, while also operating individually to provide local motor control. The distributed system is connected by a CAN network. In addition, the robot requires various sensor amplifiers and power conversion circuits.

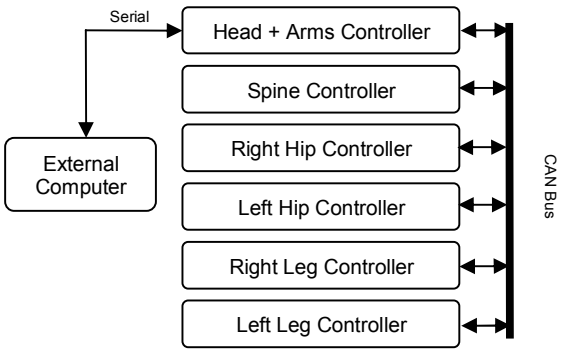

Figure 2: Block diagram of the distributed control system.

2.3 Sensors

The GuRoo currently has encoders on each of the high powered DC motors, able to provide rotational position to a resolution of 0.001 of a degree. An inertial measurement unit consisting of 3 axis rate gyroscopes and 3 axis accelerometers has been obtained that is currently being integrated into the system. Provision has been made for the future inclusion of pressure sensors on the soles of the feet and a stereo vision system.

2.4 Software

The software consists of four main entities: the global movement generation code, the local motor control, the low-level code of the robot, and the simulator [McMillan, 1995]. The software is organised to provide a standard interface to both the low-level code on the robot and the simulator. This means that the software developed in simulation can be simply re-compiled to operate on the real robot. Consequently, the robot needs a number of standard interface calls that are used for both the robot and the simulator. Figure 3 shows modularisation of the software, and the common interfaces.

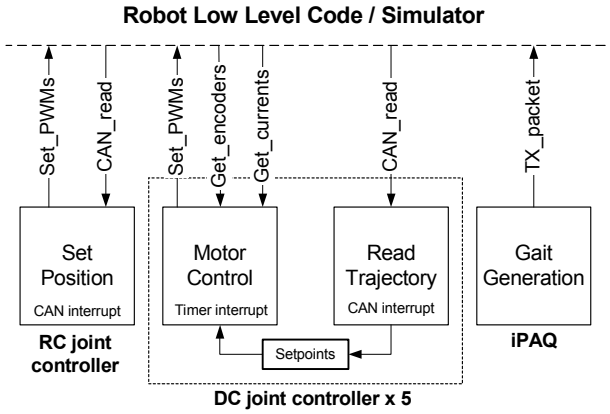

Figure 3: Block diagram of common software modules and the interface used to both the real robot and the simulator.

3 The CMAC Neural Network

The Cerebellar Model Articulated Controller or CMAC, was first described by Albus [1975]. The CMAC network can be viewed as a number of lookup tables. Each table, or Association Unit (AU), has the dimensions equal to the number of input variables. Inputs to the system are quantized and scaled to create a global lookup address.

This address is mapped to a coarser address space in each AU where a weight is stored. The AUs are structured such that a single resolution change in one input signal will result in only one different weight chosen. The output signal is calculated by finding the sum of the weights of all AUs at this lookup address. As the output result is the sum of all association unit's weights, a greater number of association unit's results in a system that is better able to generalize the input space.

The input space is dominated by hyperplanes of plausible input combinations, with large empty spaces in each AU where real-life input combinations are not physically possible.

Hashing techniques are used to reduce the memory requirements by mapping the global address space to a smaller, virtual, address space. The *modulo* function is the most simple way of achieving this. Hash collisions occur when two or more global address' hash to the same virtual address. This is not necessarily fatal, as a large number of AUs will ensure the table weight in question to have a minor effect on the overall output.

Table weights are updated using the following rule:

$$w_{new} = w_{old} + \frac{\alpha}{n}\left(\theta_{des} - \theta_{act}\right)$$

where:

w_{new} : New weight value
w_{old} : Original weight value
α : Learning rate
n : Number of association units
θ_{des} : Desired joint position
θ_{act} : Actual joint position

As the output of the response of the network is the sum of the selected table weights, the change in weight between iterations is divided by the number of association units to ensure the learning rate has the same effect regardless of the number of AUs.

4 Trajectory Error Learning

Trajectory Error Learning (TEL) is a biologically inspired method of robot motion compensation where learning is driven from the difference between the intended trajectory of the robot and the actual trajectory measured by the feedback sensor (possibly after some sensory delay) [Collins, 2003]. This section illustrates how TEL can be

applied to improve tracking performance in a humanoid robot.

4.1 Joint Position Error

As can be seen in Figure 4, during the single support phase ($2 < t < 4$), the joints are heavily loaded and experience significant position error. Conversely, during the swing phase each joint maintains positions adequately. It is these significant variations in load that prevent the PI control loop implemented on each of the GuRoo's joints from obtaining a satisfactory response. Tests with gain scheduling techniques have, as yet, to provide any improvement in performance [Roberts *et al*, 2003].

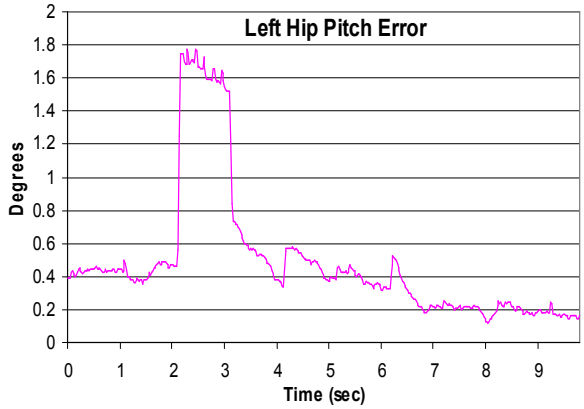

Figure 4: Graph of the position error experienced in the left hip pitch joint over one complete walking cycle. The sudden increase in error after 2 seconds relates to single support phase of the gait.

TEL uses a CMAC network to supply a compensating signal to eliminate this position error. As a typical walking gait of a humanoid is periodic in nature any errors experienced by the robot are also typically cyclic in nature: for example, the joint error during the support phase. By observing the gait phase, the CMAC learns which parts of the gait require compensation.

4.2 System Implementation

The method of compensating the joint error is illustrated in Figure 5. The trajectory of the limb is expressed as a stream of desired joint positions which are generated by the gait generator. As the motion of the robot is periodic, the state of the trajectory can be expressed as the gait phase. The gait phase is implemented as a periodic counter, incrementing every control loop and resetting at the beginning of each motion cycle.

The desired joint position is augmented by the output of the CMAC, and passed to the feedback joint controller. The inputs to the CMAC consist of the gait phase and the measured joint position, where the measured joint position will be subject to some delay with respect to the desired joint position. In this form, the CMAC is used as a predictive modulator; seeking to eliminate the error it expects to see based on the errors that it has already seen at the same point in previous cycles. The sawtooth wave of the gait phase gives the CMAC the point in the cycle that it is currently compensating, while the measured joint position accounts for different disturbance conditions that may occur at that point in time.

The error in joint position is used to train the CMAC network. A history of previous desired position commands is kept to compensate for the sensory delay experienced in the system. This history buffer also ensures weight updates are performed with the correct time delayed error.

The error signal used to train the CMAC is as follows:

$$e_k = \theta_{des(k-\tau)} - \theta_{act(k)}$$

where:

e_k	=	Error training signal(k)
$\theta_{des(k-\tau)}$	=	Desired Joint Position($k-\tau$)
$\theta_{act(k)}$	=	Actual Joint Position(k)
τ	=	Sensory Delay

Thus weights are updated τ control loops after they are used.

5 Crouching Experiment

The initial experiments that have been conducted using this method have been based on a slow crouching motion run

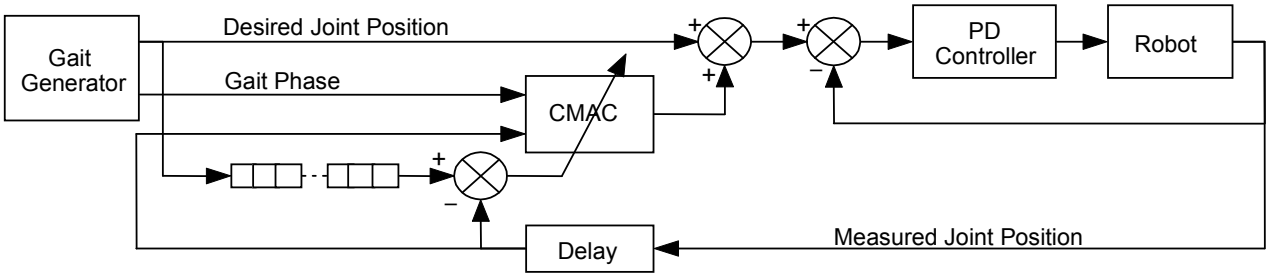

Figure 5: System diagram. The Desired Joint Position is time delayed when calculating position error to account for sensor delay. Table weights are updated a set number of control loops after being used. This delay is equal to the sensory delay inherent in the system.

over a period of 12 seconds. The pitch axis motors of the hip, knee and ankle joints follow a synchronised sinusoidal profile with a magnitude of 16, 35 and 22 degrees respectively to reach the bottom of the crouch position. This test exposes the joints to a range of dynamic and static loads, and be repeated many times without moving the robot around the laboratory.

For this experiment, the following CMAC parameters were chosen. The number of receptive units and field width were chosen to provide the necessary discrimination, while also providing local generalisation. The hashing ratio was chosen to reduce memory requirements while still keeping a low probability of hashing collisions. The learning rate was tuned to provide rapid learning of the compensation, without learning from noise. The measured joint positions were subject to a delay of 60 ms, which corresponds to a delay of 3 control cycles.

Table 2: CMAC Parameters used for learning during the crouching experiment.

CMAC Parameter	Value
Joint Position receptive units	9000
Gait Phase receptive units	1200
Field width	50
Virtual SSDs	216204
Implemented SSDs	10001
Learning rate	0.001

5.1 Existing Control

Each degree of freedom utilises a PI control loop on joint velocities which corresponds to PD control in position. Both the Proportional and Integral constants were determined by running a genetic algorithm, with a fitness function minimising trajectory error and maximising joint smoothness [Roberts *et al*, 2003].

Without TEL each joint experiences the error in position seen in Figure 6. As the crouching motion is cyclic, these errors experienced do not change from cycle to cycle and are dependent on the current phase of gait, with larger errors present in the second half of the cycle as the robot accelerates itself upwards. This displays the inability of the existing control to provide a consistent response over the whole motion cycle.

The position error is roughly cyclic, as similar errors occur at similar points during the gait phase, where gait in the context of this experiment refers to phase of the crouch. When the TEL network is enabled, position error is quickly minimised. Figure 7 and Figure 8 show the position error of the left hip pitch and the compensating signal respectively. As the error signal reduces, the amount of learning also reduces, and the change in table weights decreases. The compensating signal then becomes cyclic in nature and does not change until additional error develops. The error reduces from peak values of 0.4 degrees to the noise floor with peaks of 0.1 degrees.

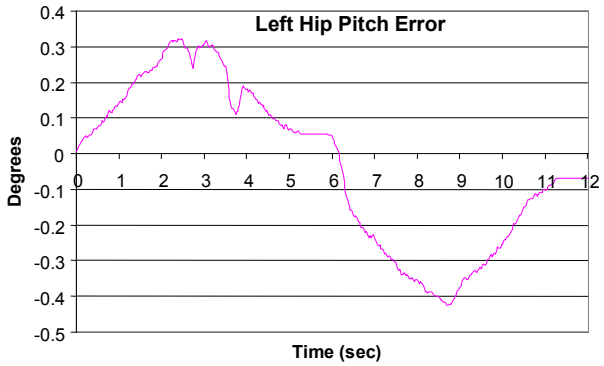

Figure 6: Position error of the left hip pitch during a crouching motion. This signal is used to drive the learning in the CMAC network. Note the larger magnitude of error during the second half of the motion, as the robot accelerates itself upwards against gravity.

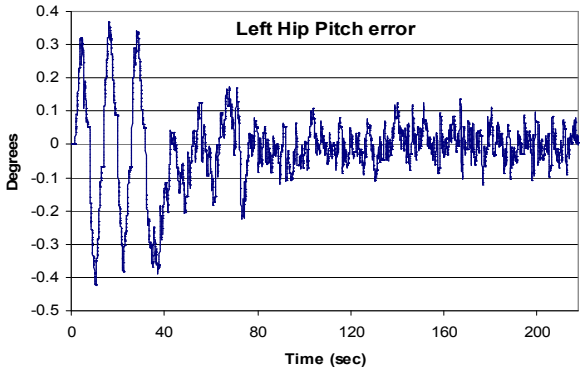

Figure 7: Position error for the left hip pitch during the crouching motion. With the Joint Compensation system in place, the maximum error experienced by the joint is reduced by 75%.

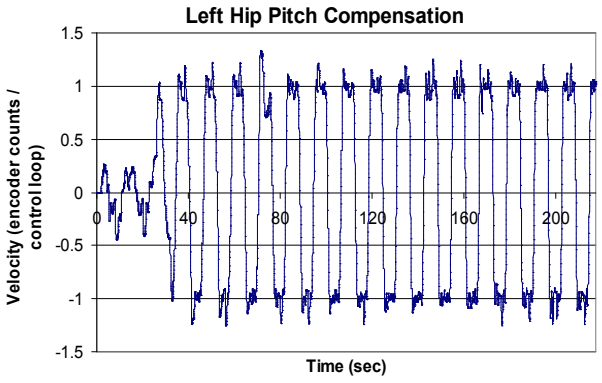

Figure 8: Compensation signal of the Left Hip Pitch. As the error signal decreases, the rate of learning also decreases and the compensation signal takes on a constant periodic form.

Similar results were obtained from all pitch movements involved in the crouching motion. Table 3 shows the increase in tracking performance for each of the joints. It can be seen that a similar performance increase was obtained on all six joints involved in the motion.

Table 3: Summary of peak error in position for all joints before and after learning.

Joint	Peak Before	Peak After	Reduction
Left Hip	0.4°	0.1°	75%
Right Hip	0.4°	0.1°	75%
Left Knee	0.6°	0.16°	73%
Right Knee	0.6°	0.14°	77%
Left Ankle	0.4°	0.1°	75%
Right Ankle	0.4°	0.1°	75%

6 Conclusions

Feedback control techniques alone are in unsuitable for control of a humanoid robot. The extreme differences in load throughout the gait, and particularly during the swing phase versus the single support phase, make feed-forward compensation necessary. Modelling the plant dynamics of a mobile body with so many degrees of freedom is a difficult task. Furthermore, interaction with an unpredictable environment and changes in payload further increase the complexity of the system dynamics.

The simple crouching experiment demonstrates the existing control loop inability to compensate for changes in load as a result of gravity. Using the error signal generated from the desired joint position and the actual joint position – the trajectory error – the cerebellar system is able to learn a response capable of decreasing the peak error by 75%. The experiments were conducted with a crouching motion on a real 23 degree of freedom humanoid and show marked reduction in position error of all joints with the implementation of the TEL system.

6.1 Further Work

In this implementation, suitable compensation of position error has been achieved for a crouching motion. It can obviously be trialled on a walking gait for improvement of walking performance.

The TEL system used as the basis of this work is suited to any control problem where a tracking error is present. Within a humanoid robot, there are many trajectories that can be used to enhance stability. Torso inclination, location of the Zero Moment Point and centre of foot pressure all follow a 'desired' path. Deviations to this path can be measured and a trajectory error calculated. This error can be used to train a separate TEL structured CMAC to improve balance and walking gaits.

7 References

[Albus, 1971] J. S. Albus, "A Theory of Cerebellar Function" *Mathematical Biosciences,* 10, pp. 25-61, 1971

[Barto, 1999] A, Barto, "Learning to reach via corrective movements", Self-Learning Robots III Brainstyle Robotics, pp. 6/1, 1999

[Roberts et al, 2003] J. Roberts, G. Wyeth, D. Kee, "Generation of humanoid control parameters using a GA", Submitted to this publication, 2003

[Collins, 2003] D. Collins, "Cerebellar Modeling Techniques for Mobile Robot Control in a Delayed Sensory Environment", PhD Dissertation, University of Queensland, 2003

[Collins and Wyeth, 2000] D. Collins, G. Wyeth, "Fast and accurate mobile robot control using a cerebellar model in a sensory delayed environment", Intelligent Robots and Systems, pp. 233-238, 2000

[Fagg et al, 1997] A. Fagg, N. Sitkoff, A. Barto, J. Houk, "A model of Cerebellar Learning for Control of Arm Movements Using Muscle Synergies", Computational Intelligence in Robotics and Automation, pp. 6-12, 1997

[Kee et al, 2003] D. Kee, G. Wyeth, A. Hood, A. Drury, "GuRoo: Autonomous Humanoid Platform for Walking Gait Research", Autonomous Minirobots for Research and Edutainment, 2003

[McMillian, 1995] S. McMillan, "Computational Dynamics for Robotic Systems on Land and Underwater", PhD Dissertation, Ohio State University, 1995.