Intelligent Motion Control with an Artificial Cerebellum

Russell L. Smith
July 1998

A thesis submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Engineering.

The department of Electrical and Electronic Engineering,
University of Auckland,
New Zealand

The Road goes ever on and on
Down from the door where it began.
Now far ahead the Road has gone,
And I must follow, if I can,
Pursuing it with eager feet,
Until it joins some larger way
Where many paths and errands meet.
And whither then? I cannot say.

—J.R.R. Tolkien

TO MY FRIENDS AND LAB-MATES —
AARON, BRIAN, CHRIS, DAVE,
DOMINIC, GAVIN, GEOFF, LEE,
MELISSA, RUSS M., STEVE, SYLVIA,
TONY AND WOEI . . .
. . . IT'S BEEN A LONG JOURNEY,
BUT THANKS TO YOU, A FUN ONE.

TO MY FAMILY, MUM, DAD AND RAEWYN.

TO MY SUPERVISOR, GEORGE.

Abstract

This thesis describes a novel approach for adaptive optimal control and demonstrates its application to a variety of systems, including motion control learning for legged robots. The new controller, called “FOX”, uses a modified form of Albus’s CMAC neural network. It is trained to generate control signals that minimize a system’s performance error. A theoretical consideration of the adaptive control problem is used to show that FOX must assign each CMAC weight an “eligibility” value which controls how that weight is updated. FOX thus implements a kind of reinforcement learning which makes it functionally similar to the cerebellum (a part of the brain that modulates movement). A highly efficient implementation is described which makes FOX suitable for on-line control.

FOX requires a small amount of dynamical information about the system being controlled: the system’s impulse response is used to choose the rules that update the eligibility values. A FOX-based controller design methodology is developed, and FOX is tested on four control problems: controlling a simulated linear system, controlling a model gantry crane, balancing an inverted pendulum on a cart, and making a wheeled robot follow a path. In each case FOX is effective: it associates sensor values with (and anticipates) the correct control actions, it compensates for system nonlinearities, and it provides robust control as long as the training is comprehensive enough.

FOX is also applied to the control of a simulated hopping monoped, and a walking biped. FOX learns parameters that fine tune the movements of pre-programmed controllers, in a manner analogous to the cerebellar modulation of spinal cord reflexes in human movement. The robots are successfully taught how to move with a steady gait along flat ground, in any direction, and how to climb and descend slopes.

Contents

[1] This appendix is derived from the author's paper in the Proceedings of the 1995 ANNES conference [114].

Nomenclature

Acronyms

CMAC	**C**erebellar **m**odel **a**rticulation **c**ontroller.
FBE	**F**eed**b**ack **e**rror.
FOX	**F**airly **o**bvious e**x**tension (to the CMAC).
IFC	**I**nternal **f**eedback **c**ontroller.
WIB	**W**eight **i**ndex **b**uffer.

CMAC symbols (introduced in Chapter 3)

y_i	Inputs.
x_i	Outputs.
z_{ij}	Feature detecting neurons (2-layer example).
a_{ij}	Association neurons (2-layer example).
n_y	Number of inputs y_i.
n_x	Number of outputs x_i.
n_a	Number of association neurons activated per input, or the number of AU (association unit) tables.
n_v	Number of association neurons a_{ij}.
n_w	Total number of physical weights per output.
w_{ijk}	Weights (neural network model).
d_j^i	Displacement for AU i along input y_j.
q_i	Quantized version of input y_i.
p_j^i	Table index of AU i along axis y_j.
α	Learning rate.
e_i	Error signal for output x_i.
f_v	Virtual address generating function.
f_h	Hashing function.

n_p	Number of physical weights.
$\min_i$, $\max_i$	Minimum and maximum values for y_i.
res_i	Input resolution; the number of quantization steps for input y_i.
μ_i	Index into the weight tables for AU i.
$W_j[i]$	Weight i in the weight table for output x_j.

FBE symbols (introduced in Chapter 4)

$y(t)$	System state.
$x(t)$	Control force (controller output, system input).
$e(t)$	Error signal.
$y_d(t)$	Desired system state over time.
$F(y, x)$	Controlled system function.
A	Trainable controller.
Q	Fixed feedback controller.
x_a	Output of A.
x_q	Output of Q.
w	Weights which parameterize A.
p	Position of the simple 2nd order example system.
p_d	Desired value of p.

FOX symbols (introduced in Chapter 5 and Chapter 6)

$F(\mathbf{y}_i, \mathbf{x}_i)$	The system function, which takes the current state and control input and generates the next time step's state.
F^*	A linear approximation to F.
$C(\mathbf{y}_i)$	The "critic" function, which generates a scalar error value at each time step.
$\mathbf{y}_i$	The system state vector for time i. Size $n_y \times 1$.
$\mathbf{x}_i$	The system control input vector for time i (the output of the CMAC controller). Size $n_x \times 1$.
$\mathbf{s}_i$	A vector of CMAC association-unit "sensors" that encodes the current system state. Size $n_s \times 1$.
W	The matrix of CMAC weights. Note that $\mathbf{x}_i = W\,\mathbf{s}_i$. Size $n_x \times n_s$.
e_i	A scalar error value generated by the critic function for time i.
y_i^d	A scalar position reference (for time i) used in the calculation of e.

E	The total scalar error of the entire system.
w_{pq}	Element (p,q) of the matrix W.
ξ_i^{pq}	The eligibility vector for weight w_{pq}. Size $n_y \times 1$.
$C\xi$	The eligibility profile (scalar).
$\hat{\mathbf{s}}_i^{pq}$	A synonym for $\mathrm{d}\mathbf{x}_i/\mathrm{d}w_{pq}$. $\hat{\mathbf{s}}_i^{pq}$ is all zero except for its p'th element which is equal to the q'th element of $\mathbf{s}_i$. Size $n_x \times 1$.
n_w	Total number of weights, $n_w = n_s\, n_x$.
n_a	Number of association units in the CMAC, i.e. the number of nonzero elements in $\mathbf{s}_i$.
A	A system matrix for the linear system F^*. Size $n_y \times n_y$.
B	A system matrix for the linear system F^*. Size $n_y \times n_x$.
C	The matrix for the simple linear critic function. Size $1 \times n_y$.

FOX learning rate symbols

α	The main FOX learning rate (scalar).
β	The output limiting FOX learning rate (scalar).
γ	The output derivative limiting FOX learning rate (scalar).
α_1, α_2	The overshoot error function FOX learning rates (scalars).

Eligibility profile cookbook symbols

k_a, k_b	Parameters for the selection of over-damped second order eligibility profiles (see Appendix E).
$t_{\max}$	Parameter for selecting critically damped second order eligibility profiles (see Appendix E).

FOX algorithm symbols

σ	Cut-off (decay-to-zero) point for the eligibility profile.
δ	WIB buffer size, $\delta = \sigma + 2$
λ_i	The WIB buffer, an array of δ vectors each of size $1 \times n_y$.
T_t	The current time step.
g	The position of the current time step in the WIB.
T_0	The time step at λ_0.
Λ	The accumulated value of: $e\, CA^i$ (size $1 \times n_y$).
Γ	The current value of: CA^g (size $1 \times n_y$).

Contents of the CDROM

This thesis is accompanied by a CDROM containing several movies in MPEG and AVI format. The subdirectory `thesis` on the CDROM also contains postscript files for this thesis (individual chapters as well as the entire document).

Inverted Pendulum Movie (AVI)

- `inverted.avi` : This shows various stages of the inverted pendulum experiment:
  - PD controller with a zero reference.
  - Trained FOX controller with a zero reference.
  - Trained FOX controller with a square wave reference.
  - PD controller with reference $= \sin(x) + \cos(2x)$.
  - Trained FOX controller with reference $= \sin(x) + \cos(2x)$.

Hopping Robot Movies (MPEG)

- `hopper1.mpg` : This shows four out-takes from the training of the hopping robot. Watch it fall over in various ways as it learns.

- `hopper2.mpg` : This shows the fully trained hopping robot following a path through its environment. It can go up and down a ramp without falling over.

- `hopper3.mpg` : This shows one of the hopping robot's failed attempts to climb the ramp. A foot trail is rendered, which can help diagnose the problem.

Biped Walking Movies (MPEG)

- `walk1.mpg` : Only the motion sequencer is used in the controller, no other controller modules are active. The biped makes stereotyped stepping movements and falls down immediately.

- `walk2.mpg` : The effect of hip side compensation is demonstrated. Before hip side compensation training the biped sways from side to side as it walks. After training the biped stays more upright.

- `walk3.mpg` : The combined effect of hip twist compensation and arm swing is demonstrated. Before these things are trained the biped twists from left to right as it walks. After training the biped is steadier.

- `walk4.mpg` : The effect of global side drift compensation is demonstrated. Before compensation training the biped becomes unstable when it leaves its desired path.
- `walk5.mpg` : A tripping event is demonstrated — the biped's foot touches the ground prematurely during a walking cycle.
- `walk6.mpg` : The effect of leg placement is demonstrated. Without it the biped makes no attempt to correct a falling motion. With correct leg placement the biped will place the feet to try and keep the body upright.
- `walk8.mpg` : Some out-takes from training are shown — the biped falls over in various ways as it learns to walk.
- `walkt1.mpg`, `walkt2.mpg`, `walkt3.mpg` : Successful walking — walking with a steady gait along flat ground, walking at various speeds, changing direction, and walking up and down slopes. Three copies of the movie are provided which use different MPEG bit rates (there are differences in movie quality).
- `explode.mpg` : This shows what can happen when the simulation goes astray: an incorrectly configured joint controller produces a large force that sends the robot flying into the air.