A neural network methodology is developed for air-to-fuel (A/F) ratio control of automotive fuel-injection systems. The dynamics of internal combustion engines and fuel-injection systems are extremely nonlinear, impeding methodical application of control theories. Thus, the design of standard production controllers relies heavily upon calibration and look-up tables. A neural network-type controller is developed in this article for its function-approximation abilities and its learning and adaptive capabilities. A cerebellar model articulation controller (CMAC) neural network is implemented in a research automobile to demonstrate the feasibility of this control architecture. Experimental results show that the CMAC fuel-injection controller is very effective in learning the engine nonlinearities and in dealing with the significant time-delays inherent in engine sensors.

Minimizing Emissions

The objective of a fuel-injection controller is to maintain the air-to-fuel (A/F) ratio at a value desirable for driveability and emission minimization. Fig. 1 depicts the efficiency of the catalytic converter for oxides of nitrogen (NOx), hydrocarbons (HC), and carbon monoxide (CO). For maximization of efficiency, and thus minimization of harmful emissions, the desirable A/F ratio is the stoichiometric mass ratio, which is taken as 14.65:1.

In the past, the performance of the production engine control module (ECM) has been sufficient to pass all government-mandated emissions standards and to provide adequate drive feel. However, recent legislation will soon require a much cleaner engine. By the year 2003, laws such as the 1988 California Clean Air Act require 10% of a manufacturer’s fleet to be zero-emission vehicles (ZEVs), an 84% decrease in HC emissions, a 74% decrease in NOx output and a 60% reduction in CO production for the entire fleet [2]. These regulations will soon require extensive improvements to the current fuel-injection controllers.

The standard ECM controller is based on a proportional-plus-integral-type model with a large array of varying gains stored in a look-up table generated from calibration and tuning processes. To simplify this, the use of the variable structure control (VSC) method for fuel-injection systems was proposed and experimentally demonstrated in [3]. A neural network method for fuel-injection systems is proposed in this article as an alternative. The CMAC neural network was selected for its efficient computation and fast learning [4]. In this article, the feasibility and potential for CMAC to control fuel-injection systems in real-time are demonstrated using a research automobile.

CMAC Background

A neural network is patterned after the functioning of the human brain. As such, most neural networks have a type of receptive field which is activated as a neuron would be in the brain or in the body. These receptive regions are arranged in a way dependent upon the network algorithm chosen so that a form of generalization is provided. That is to say that inputs which are similar in value will produce similar outputs. The strength of a neural network lies in the fact that no a priori knowledge of the dynamics of the system to be controlled is required. The neural network is trained by example; it passively accepts inputs and is given the corresponding desired output. It then learns to produce this same output using an iterative algorithm. Once the neural network has been trained on an acceptably general set of inputs, it is allowed to respond, that is, to produce its version of the output from what it has learned. Some networks allow on-line learning, that is, if a correct version of the desired output is still available, or if there is some measure of the correctness of the output, then feedback exists where the neural network can continue to learn as it is responding to a set of inputs.

The CMAC neural network provides a method of adaptive control that is appropriate for use in real-time control applications [5], [6]. The generalized CMAC algorithm is best described using a set-like diagram as shown in Fig. 2 and a series of three mappings between spaces or sets.

Presented at the 1993 IEEE International Symposium on Intelligent Control. M. Majors is with the University of Cambridge, England. J. Stori is with the University of California at Berkeley. D.-I. Cho is with the Department of Control and Instrumentation Engineering, Seoul National University, Seoul 151-742 Korea. This work was was partially supported by the Daewoo Motor Company while M. Majors and J. Stori were undergraduate Research Assistants to D.-I. Cho in the Department of Mechanical and Aerospace Engineering, Princeton University, Princeton, NJ.
As in [7], the first operation maps each input vector \( s \) in the \( \mathcal{S} \) state space into a binary selector \( a \), in the \( N_A \) dimensional conceptual memory space \( \mathcal{M} \).

\[
f: \mathcal{S} \rightarrow a = \{0, 1\}^{N_A}.
\]

The mapping \( f \) is the essence of the CMAC and is performed by quantizing the \( \mathcal{S} \) space into \( N_A \) overlapping hypercubes known as receptive regions. The \( j \)th receptive region will become active and the \( j \)th element of \( a \) will be turned on if a given input is within its span. Each input will excite exactly \( C \) receptive fields; \( C \) is known as the generalization parameter. The mapping \( f \) also provides for the property of generalization: inputs that are “close” in \( \mathcal{S} \) space will be “close” in \( \mathcal{M} \) space such that more points overlap.

As described in [7], the CMAC algorithm is most easily understood in the single input case. Here, the hypercubical receptive fields are merely line segments. The input space \( \mathcal{S} \) can be divided into \( N = (N_A - C + 1) \) quantization intervals over \( [s_{\text{min}}, s_{\text{max}}] \) using \( (N + 2C - 1) \) divisors \( \lambda_j \), where

\[
\lambda_{1-C} = \ldots = \lambda_0 = s_{\text{min}}
\]

\[
s_{\text{min}} \leq \lambda_j \leq \lambda_{j+1} \leq s_{\text{max}} , \ j = 0, \ldots, N - 1
\]

\[
\lambda_N = \ldots = \lambda_{N+C-1} = s_{\text{max}}
\]

Therefore, the \( j \)th element of the associative memory vector \( a \) is

\[
a_j = \begin{cases} 
1 & \text{for } s \in [\lambda_j - C, \lambda_j) \\
0 & \text{otherwise}
\end{cases}
\]

A single-input mapping with \( C=3 \) is shown in Fig. 3.

The number of elements required for the selector vector \( a \), with \( R_j \) as the range and \( d_j \) the quantization width, for the \( j \)th input, \( C \) the generalization parameter, and \( N \) the number of inputs, is

\[
C = \prod_{j=1}^{N} \left( \frac{R_j}{d_j C} \right).
\]

While the memory required for physical implementation of the CMAC decreases with a greater generalization parameter, it increases dramatically with decreased quantization width and increased dimensions. The generalization parameter \( C \) is the ratio of the width of the receptive fields to the offset between adjacent fields. If the quantization widths are held fixed, a larger generalization parameter will result in a CMAC which requires less training to produce comparable responses to new data. An increase in generalization capability results in loss of precision.

The possibility of a prohibitively large associative memory vector may require a second mapping. The second mapping \( \mathcal{R} \) goes from the conceptual memory space \( \mathcal{M} \) to the physical memory space \( \mathcal{D} \):

\[
\mathcal{R}: \mathcal{M} \rightarrow \mathcal{D}.
\]

The mapping \( \mathcal{R} \) can be one-to-one if a large memory requirement can be satisfied. To reduce the necessary memory a hash coding that maps a large \( \mathcal{M} \) to a smaller \( \mathcal{D} \) can be used as in [5] and [6]. In our controller implementation, the memory available in a 386 PC proved sufficient and hashing was not needed. The third mapping goes from the selector vector \( a \) to the scalar output \( y \):

\[
y: \mathcal{D} \rightarrow \mathcal{Y}
\]

performing an inner product with the \( N_A \) dimensional weight vector \( w \):

\[
y = w^T a.
\]

Since \( a \) is binary, this is a summation of the \( C \) weights corresponding to the active receptive fields.

The training of the CMAC is accomplished through the update of the weight vector such that

\[
w_j = w_{j-1} + \frac{\xi (y_d - y)}{C} m
\]

where \( y \) is the output of the CMAC response to a given input, \( y_d \) is the desired response to the same input, \( C \) is the generalization parameter and \( \xi \) is a learning rate with a value between 0 and 1.
The mappings described above depict a neural network with three key features: the generalization parameter $C$ can be controlled independently of the number of input dimensions and the size of the physical weight space. Uniform approximation is realized through the property that a change of one quantization interval in one input will cause exactly one receptive field to become active and one to become inactive. Finally, the receptive fields are arranged in such a fashion that summations of weights throughout the space are accomplished with a minimum of calculations.

The one-dimensional CMAC depicted in Fig. 3 can easily be extended to $N$ dimensions where $N$-dimensional hypercubes are arranged in a geometrically regular pattern to achieve the desired properties. This realization can, at its most basic, be implemented as an $N$-dimensional array of weights or look-up table such that the computational requirements of the CMAC neural network are minimal and response is quick.

**CMAC Fuel-Injection Control**

Any engine state is well described by the manifold pressure (load) and the engine speed. (Detailed engine model used in this article can be found in [3] and [8] and for brevity is not repeated here.) These two inputs were chosen for the CMAC.

The CMAC neural network can be implemented very efficiently. Each of the two inputs index one dimension of a three-dimensional memory space. The third dimension corresponds to the $C$ layers of receptive fields and provides the generalization. A mapping of the first input is achieved as below, where $i$ indexes each of the $C$ layers:

$$F_i = (U1 + C - i) \div C.$$  

In the above, $i$ denotes the layer, $\div$ is integer division, and $U1$ is a quantized index of maximum value $S1 \times C$, where $S1$ is the physical memory size in the first dimension. Thus the quantization width is $input_{max}(S1 \times C)$, and precisely $C$ receptive fields will be activated. With this, the change of a single quantization width in input will result in exactly one current field becoming inactive and one new field becoming active. For the fuel-injection control problem, $C = 32$ gave a reasonable trade-off of generalization versus performance.

To learn on-line, the CMAC requires feedback from an A/F ratio sensor. Production vehicles use an oxygen ($O_2$) sensor with a switching output about the desired stoichiometric A/F ratio. Production vehicles use an oxygen ($O_2$) sensor with a switching output about the desired stoichiometric A/F ratio as shown in Fig. 4. Linear A/F ratio sensors are commercially available for laboratories, but the cost of such sensors in their present technology is prohibitive for production use.

In this article, the use of a linear A/F sensor for the initial CMAC learning is considered. Due to the expense of the linear sensor, its usage will be confined to the laboratory where a single network will be trained for installation in a production line. Then an $O_2$ sensor-based learning algorithm can be activated post-production. Since the CMAC can be well-trained initially, a low learning rate can maintain stoichiometric A/F ratio control over the life of the vehicle as the engine dynamics change due to age. In addition, research is underway for a CMAC-based controller which only utilizes the production $O_2$ sensor. The results will be reported later.

In order to reduce the training time of the CMAC, the weight space is initialized with generic values derived from the model in [8]. Based on this model, the desired mass rate of fuel to command $m_{fuel}$ for a stoichiometric A/F ratio is

$$m_{fuel} = \frac{c_1 \cdot \eta_{vol} \cdot m_a \cdot 0_k}{\beta_{stoich}}$$  

where $c_1$ is a physical constant, $\eta_{vol}$ is the volumetric efficiency, $m_a$ is the mass of air in the intake manifold, $0_k$ is the engine speed, and $\beta_{stoich}$ is the stoichiometric A/F ratio. Since $m_a$ cannot be directly measured, the ideal gas law is used to produce an expression for the desired fuel command rate as a function of $P_m$, the manifold pressure $T_m$, the manifold temperature, and the gas-constant for air $R$:

$$m_{fuel} = \frac{V_e \cdot \eta_{vol} \cdot P_m \cdot 0_k}{4\pi \cdot R \cdot T_m \cdot \beta_{stoich}^2}.$$  

In (11), $c_1$ has been expressed in terms of the engine displacement $V_e$ and the cylinder geometry. Though the manifold temperature appears directly in (11), it is not needed as an input to the CMAC since it remains relatively constant. However, $T_m$ is used as an input to the control system. A temperature-dependent set of linear correlations between $O_2$ sensor output and linear A/F sensor output, as in Fig. 4, have been determined to allow the CMAC to utilize the same learning routine for both linear A/F sensor and $O_2$ sensor.

Initially the CMAC is trained using (11) with a generic model for $\eta_{vol}$. If the volumetric efficiency $\eta_{vol}$ were known, this expression would provide the exact command for a stoichiometric A/F ratio. Other variables in (11) are either known or measured using production sensors. However, $\eta_{vol}$ cannot not be modeled analytically [3]. In fact, this model serves only to initialize the CMAC weight space since it contains large errors, resulting in poor performance. To improve the CMAC performance, the weights need to be accurately adjusted. To learn in real time, a linear A/F ratio sensor is used for feedback to learn and adjust weights in real time.
Due to the downstream location of the A/F sensor in the exhaust manifold, the feedback is significantly delayed. The CMAC weight update law (8) is developed from (11) for this application. Since the delayed response to the old command $m_{fc,old}$ is known to be $\beta_{exact}$, the current A/F ratio reading from the sensor, $m_{fc, old}$, can be expressed from (11) by substituting $\beta_{exact}$ for $\beta_{exact}$. Thus,

$$m_{fc,old} = m_{fc,old} \left( \frac{\beta_{exact}}{\beta_{exact}} \right).$$

(12)

Now algebraic manipulation yields the desired change in mass rate of fuel to command at the old state:

$$
\Delta m_{fc,old} = m_{fc,old} - m_{fc,old} \\
= m_{fc,old} \left( \frac{\beta_{exact}}{\beta_{exact}} - 1 \right)
$$

(13)

Substituting $\Delta m_{fc, old}$ from (13) for $(y_k - y)$ in (8) provides a general learning algorithm for the CMAC. This specifies an incremental change in each weight activated during the command $m_{fc,old}$ so that the next time that the same combination of engine states is achieved, the new, more desirable, command will occur.

Conventional control methodologies cannot easily deal with time-delays. The CMAC can easily compensate for the delayed information by storing past A/F ratio responses in a time dimension of the receptive field arrays as discussed above. This is one of the major advantages of the CMAC-based new fuel-injection control method developed. However, this task is complicated by the dependency of the magnitude of the time delay upon both the manifold pressure and the engine speed. Since the dynamics of the exhaust manifold are complex, an empirical calibration was used to relate the time-delay magnitude to the manifold pressure, in turn allowing the current A/F reading to modify the CMAC weights from several engine revolutions previous.

**Experimental Results**

The CMAC neural network controller has been implemented on a 1988 Oldsmobile Calais with a Quad-4 (double-overhead-cam, four-cylinder) engine. The Calais is equipped with two 386-based portable Dolch microcomputers. The Dolch computers contain RTI-815 input/output boards for data acquisition and control. One computer receives an input voltage from the throttle position sensor (TPS) and commands the throttle actuator for drive-by-wire testing. The CMAC controller is implemented on the second computer, which collects engine data and executes the CMAC control routine each engine revolution. A photograph of the test vehicle is shown in Fig. 5.

The control objective is to maintain a stoichiometric A/F ratio, with deviations limited to a maximum of $\pm 1\%$. This stringent control requirement is derived from the degradation of the catalytic converter efficiency by as much as 50% for deviations from stoichiometry of 1%, as seen in Fig. 1. Thus the controller will maximize the catalytic converter efficiency and thereby meet emission standards. The CMAC fuel-injection controller performance has been assessed on two driving scenarios. All appropriate efforts were made to assure the uniformity of the states for each controller evaluation. After initializing the CMAC weight space with a generic model of the volumetric efficiency using (11), the testing for each scenario proceeded linearly and independently with learning on all trials. The data were collected with the test-vehicle in “drive” (automatic transmission) as it would likely be operated under normal driving conditions. All testing has utilized the linear A/F sensor; future work will address the usage of the O2 sensor routines. The results demonstrate the ability of the CMAC to maintain a stoichiometric A/F ratio in a controlled environment.

The first test involves the car slowly accelerating from a stop. As seen in Fig. 6, the ECM provides accurate control of the A/F ratio, nearly keeping it within the desired $1\%$ boundaries. The exception is the rich (12.4 A/F ratio) peak for the 9.1-14.9 second period. This perturbation is difficult for us to explain but occurred frequently during ECM testing.

The performance of the CMAC controller with the linear A/F sensor is depicted in Fig. 7. For this test, the learning rate was set to $\xi=0.5$. The CMAC maintains the A/F ratio within the $\pm 1\%$.
boundary for the entire duration of the test. The CMAC performance is better than that of the ECM in this test. Since the performance of the CMAC is now acceptable for this test, the learning may now be halted. With this fixed set of weights and without further learning, the CMAC will continue to achieve the same good results on more trials of the same test. Thus, when the CMAC has been fully trained to produce stoichiometric A/F ratio control over a large set of possible engine states, the feedback from the A/F sensor is no longer necessary and the CMAC could perform open-loop control with acceptable results.

To demonstrate the ability of the CMAC to learn stoichiometric control of a more difficult scenario, the double tip-in/tip-out driving test shown in Fig. 8 is evaluated. The performance of the ECM controller is shown in Fig. 9. At tip-in at \( t = 6.0 \) s, the ECM has a lean spike of magnitude A/F ratio 17.1. This lean region leads to a rough acceleration. This occurs again at the second tip-in at 18.0 s. The ECM is able to maintain the A/F ratio within the ±1% boundaries only during the first steady-state region from 0 to 6 s. Even during the quasi-steady-state conditions from \( t = 7-11 \) s and \( t = 17-23 \) s, the ECM commands saw-tooth-like rich regions with A/F ratios as low as 11.8. At the low pressure region at the end of test, after 25 s, the ECM goes extremely lean (>20). The poor performance displayed in Fig. 9 was recorded contemporaneously with the following CMAC results and will therefore be used as a standard for comparison.

Fig. 8. Second throttle scenario.

Fig. 9. ECM control on second scenario.

Fig. 10 shows the performance of the CMAC fuel-injection controller during open-loop control only after training by the generic expression for the volumetric efficiency. The performance is very poor. However, after four iterations of the linear A/F sensor learning routine at a learning rate of \( \xi = 0.5 \), the performance is improved to that shown in Fig. 11. The CMAC-controlled A/F ratio goes lean briefly at tip-in \( (t = 6, 18 \) s) and more significantly rich at tip-out \( (t = 12, 24 \) s). Other than during these short periods, near stoichiometry is achieved. The performance improvement is drastic and in general much better than the production ECM control.

Fig. 10. Open-loop CMAC control on second scenario.

Fig. 11. CMAC control on second scenario, fourth trial.

These two limited testing scenarios have demonstrated that with the linear A/F ratio sensor learning routine, the CMAC-based controller is capable of improving upon the performance of the ECM controller over a certain band of manifold pressures and engine speeds. It is expected that with training over a suitably generic set of inputs, the CMAC will be able to maintain a nearly stoichiometric A/F ratio over all driving conditions. This performance should be adequate even without further learning so that the trained CMAC could be used in open-loop control without further on-line learning. The use of the binary A/F ratio sensor need not necessarily be implemented but to regulate performance (i.e., adapt to changes in engine dynamics) over the lifetime of the vehicle.

Conclusions

A CMAC neural network controller has been designed for a fuel-injection system and implemented in a laboratory test vehicle. The designed CMAC controller is very computationally efficient, and therefore suitable for real-time implementation. While only the expensive linear A/F ratio sensor learning routine was implemented, the results demonstrate that the CMAC controller has excellent potential for good emissions performance. The continuation of reported work shows that the CMAC method can be effectively implemented in production vehicles with only the standard binary A/F sensor. These results will be reported later.

References


**James Stori** received the B.S.E. degree in mechanical and aerospace engineering from Princeton University in 1993. He is currently pursuing the Ph.D. degree in mechanical engineering at the University of California at Berkeley where his research interests include neural networks, process planning, and intelligent control for rapid-prototyping machining application. He is a member of Tau Beta Pi and Sigma Xi.

**Dong-II Dan Cho** received the B.S. degree from Carnegie Mellon University and the S.M. and Ph.D. degrees from the Massachusetts Institute of Technology. He was an Assistant Professor of Mechanical and Aerospace Engineering at Princeton University from 1987 to 1993. Since 1993, he has been with the Department of Control and Instrumentation Engineering in the School of Electrical Engineering at Seoul National University. His research interests are nonlinear control, mechatronics, automotive control and microsensors and actuators. He is a member of IEEE, KIEE, KITE, and ASME and serves as an Associate Editor for the IEEE/ASME Journal of Microelectromechanical Systems and the IOP Journal of Micromechanics and Microengineering.