CONTINUOUS ACTION REINFORCEMENT LEARNING APPLIED TO VEHICLE SUSPENSION CONTROL

M. N. HOWELL, G. P. FROST, T. J. GORDON and **Q.H. WU+.**

Loughborough University, Department of Aeronautical and Automotive Engineering and Transport studies.
+ University of Liverpool, Dept. of Electrical Engineering and Electronics.

Abstract - A new reinforcement learning algorithm is introduced which can be applied over a continuous range of actions. The learning algorithm is reward-inaction based, with a set of probability density functions being used to determine the action set. An experimental study is presented, based on the control of a semi-active suspension system on a road going, four wheeled, passenger vehicle. The control objective is to minimise the mean square acceleration of the vehicle body, thus improving the ride isolation qualities of the vehicle. This represents a difficult class of learning problem, owing to the stochastic nature of the road input disturbance together with unknown high order dynamics, sensor noise and the non-linear (semi-active) control actuators. The learning algorithm described here operates over a bounded continuous action set, is robust to high levels of noise and is ideally suited to operating in a parallel computing environment.

1. INTRODUCTION

In this paper reinforcement learning is used on-line to improve the ride performance of an experimental vehicle fitted with controllable suspension dampers (semi-active suspension). A teaching signal, or critic, provides a scalar evaluation signal representing the learning system performance. This is used to update the learning system's internal states with the aim of improving closed-loop system performance over time.

Compared with standard mechanical 'passive' suspensions, electro-mechanical active and semi-active systems can offer significantly improved control of motor vehicle body dynamics. While active systems require a source of mechanical power, such as an hydraulic pump, semi-active systems control suspension force by dynamically varying the area in a flow control valve within the hydraulic damper unit, and are purely dissipative [1]. However, detailed controller design is far from straightforward; the dynamics of an actual vehicle becomes very complex when a realistic range of frequencies is considered, typically below 30Hz for ride comfort analysis. Factors such as longitudinal suspension compliance, body flexibility, and engine mounting degrees of freedom can be important, with the result that a model-based controller design strategy may be very time-consuming or even ineffective in practice. These considerations motivate the use of real-time learning of control system design.

Reinforcement learning benefits over other types of learning system in requiring no dedicated training and evaluation phases of learning; instead, such systems progressively adapt to the given environment. It has previously been applied in areas such as game playing [2], robotics [3] and the control of power systems [4]. However the majority of reinforcement learning algorithms available (e.g. Q-learning, adaptive heuristic critics and learning automata) are based on discrete action sets, which is very restrictive for the present dynamic control problem. Although several methods have been developed to overcome this limitation, such as generalised approximation techniques [5] (where the generalisation replaces costly training experience) and homing in techniques [6], there remains a need for a method that can be applied over a continuous action space. Reinforcement learning is a form of machine learning that tries to maximise a scalar reward through interaction with a stochastic environment. The reward is more evaluative than instructional and indicates only the degree of success the actions achieved in the environment but not which action would

have been optimal. This differs in a fundamental way from supervised learning situations in which the learning system is typically shown a series of input-output examples of which is has to match.

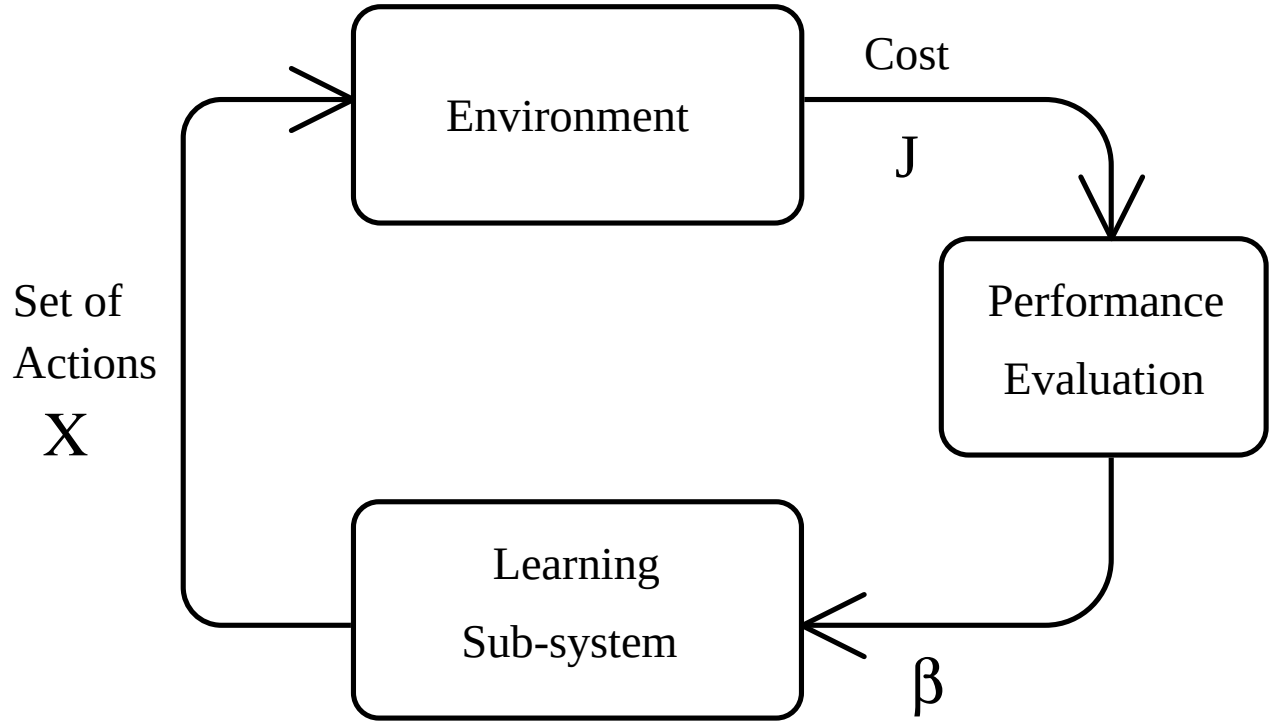

Fig. 1. Reinforcement Learning System.

In Fig. 1 a typical reinforcement learning system is shown. The learning sub-system sends an action or set of actions to the environment which then returns a scalar value, via the performance evaluation function, indicating the quality of that action. The performance evaluation function encodes the explicit goals and objectives of the learning system and returns a scalar value, β, indicating the quality of the applied actions, where 1 indicates maximum reward and 0 indicates a null reward. The environment is stochastic, so actions have to be applied many times in order to determine which action (or actions) yield the highest mean reward from the environment. With all reinforcement learning systems a balance has to be maintained between exploiting the information gained during learning, and exploring the set of actions to gain more information.

The stochastic Continuous Action Reinforcement Learning Automata (CARLA) algorithm described below was developed as an extension of the learning automata approach [7, 8] and can be applied across a continuous range of actions. The algorithm is of a reward inaction type with multiple actions being implemented in a similar manner to interconnected learning automata, where the interconnection is through the dynamics of the 'environment', in this case the vehicle. The general requirements for the CARLA algorithm are that it should operate in stochastic environments, be used on-line and be adaptable to changing environmental conditions.

2. DISCRETE-ACTION LEARNING AUTOMATA

Stochastic learning automata, are a form of non-associative reinforcement learning that operate on a finite discrete action set $\{x_1, x_2, \cdots x_r\} \in X$ with randomised action selection. Each action x_i has a probability p_i of selection associated with it. It is assumed that no information about the actions is available at the start of learning and therefore the selection probabilities are initially equal. The actions are selected on the basis of the probability distribution, which are iteratively updated based on a performance evaluation signal β received. Many different learning rules have been developed with varying convergence properties; for a description of these see [8]. One of the algorithms which has been shown to have good convergence properties is the linear reward-inaction algorithm described below. In response to action x_i being selected, at time step n, the probabilities are updated as given in eqn 1.

$$\begin{aligned} p_i(n+1) &= p_i(n) + \theta\beta(n)\big(1 - p_i(n)\big) \\ p_j(n+1) &= p_j(n) + \theta\beta(n)p_j(n) \qquad \text{if} \quad i \neq j \end{aligned} \tag{1}$$

Here θ is a learning rate parameter, $0 < \theta < 1$ and $\beta \in [0,1]$ is the value indicating the quality of reward received for the action, 1 indicating maximum reward and 0 indicating a null reward.

A single learning automaton is a relatively simple unit but can be used a building block for more complex systems with multiple actions. The effectiveness of the method derives from the interconnection of many of the automata which interact through the environment. This makes them suitable for distributed and intelligent control applications with the automata operating in a decentralised yet co-ordinated manner. For a continuous action variable a corresponding discrete action set can be created by discretizing at regular intervals; however the speed of convergence will decreases as the number of actions increases with the initial probabilities of selection for any action being very small.

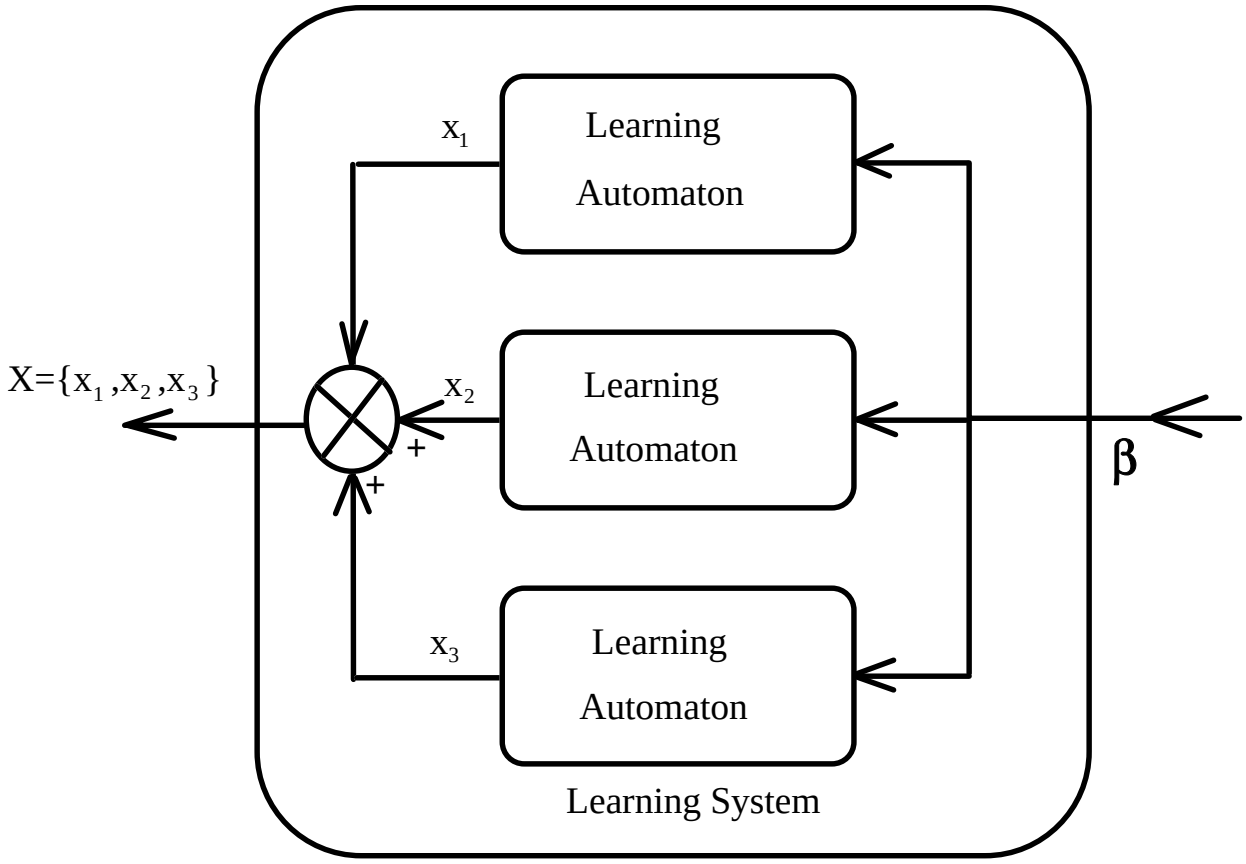

Fig. 2. Interconnected Automata

The CARLA algorithm was developed as a natural extension to the above learning automata algorithm to allow actions to be selected over a bounded continuous range.

3. CONTINUOUS ACTION REINFORCEMENT LEARNING AUTOMATA

In this section, the general formulation of the CARLA algorithm is given. This is based on the simple case of a single action being presented to the environment at each iteration, however, multiple actions will typically be configured according to Fig. 2. Let the CARLA action variable x be a bounded continuous random variable defined over the interval $X = [x_{\min}, x_{\max}] \subset \mathbb{R}$. The discrete probability distribution of Section 2 is then replaced, at iteration n (where $n = 0,1,2...$), by a continuous probability density function $f(x,n)$ which satisfies

$$\int_{-\infty}^{\infty} f(x,n)dx = \int_{X} f(x,n)dx = 1 \qquad (2)$$

and $f(x,n) \geq 0 \ \forall n, \forall x$.

The generic pseudo-code of the CARLA algorithm then takes the following form

Initialise the probability density function to a uniform distribution
Repeat
    Select an action using its probability density function
    Execute action on the environment
    Receive cost/reward for previous action
    Update performance evaluation function β
    Update probability density function
Until stopping condition

Fig. 3. Generic pseudo-code for a single CARLA

The initial distribution is chosen as uniform:

$$f(x,0) = \begin{cases} \dfrac{1}{x_{\max} - x_{\min}} & \text{for } x \in X \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

In this case, the random choice of x is obvious, $x \sim \mathrm{U}[x_{\min}, x_{\max}]$. More generally $x(n)$ needs to be selected as a random variable based on the non-uniform distributed function $f(x,n)$. This is achieved via a uniform variable $z(n)$. Given $z(n) \sim \mathrm{U}[0,1]$ at iteration n, the value of the action is chosen so that

$$\int_{x_{\min}}^{x(n)} f(x,n)dx = z(n) \tag{4}$$

The action $x(n) = \xi$ is then applied to the environment which returns a dynamic cost or performance index $J(n)$ to the performance evaluation function that indicates the quality of the chosen action. The current and past costs are stored in a reference set *R* and used to determine the value of reward $\beta(n)$, where

$$\beta(n) = \max\left\{0, \frac{J_{med} - J(n)}{J_{med} - J_{\min}}\right\} \tag{5}$$

J_{med} and $J_{\min}$ are the median and minimum values of the stored costs in the reference set respectively. The reward $\beta(n) \in [0,1]$ relates the performance of the most recent action $J(n)$ to the results obtained for past actions. A high value of $\beta(n)$ indicates reward and a low value punishment. To avoid problems with infinite storage and to allow the system to adapt to changing environments, only the last m values of costs are stored.

Based on the performance index, the probability density function $f(x,n)$ is updated, according to the rule

$$f(x,n+1)=\begin{cases}\alpha\left[f(x,n)+\beta(n)H(x,r)\right] & \text{if } x\in X\\ 0 & \text{otherwise}\end{cases} \tag{6}$$

where α is determined implicitly by the normalisation condition

$$\int_{x_{\min}}^{x_{\max}} f(x,n+1)dx=1 \tag{7}$$

Here $H(x,r)$ is a symmetric Gaussian neighbourhood function centred on $r=x(n)$

$$H(x,r)=\lambda\exp\left(-\frac{(x-r)^2}{2\sigma^2}\right) \tag{8}$$

and λ and σ are parameters that affect the height and width of the neighbourhood function. They may be defined in terms of the range of actions, namely

$$\lambda=\frac{g_h}{\left(x_{\max}-x_{\min}\right)} \tag{9}$$

$$\sigma=g_w\cdot(x_{\max}-x_{\min}) \tag{10}$$

where the speed and resolution of the learning are controlled by the two free parameters g_h and g_w respectively. If we assume the cost function is reasonably smooth over the action space then similar actions should produce similar rewards. The function $H(x,r)$ has the effect of spreading the reward for the selected action to neighbouring actions. If convergence occurs then the distribution at a single point obtained by the CARLA will approach that of the neighbourhood function $H(x,r)$ as shown in Fig. 4.

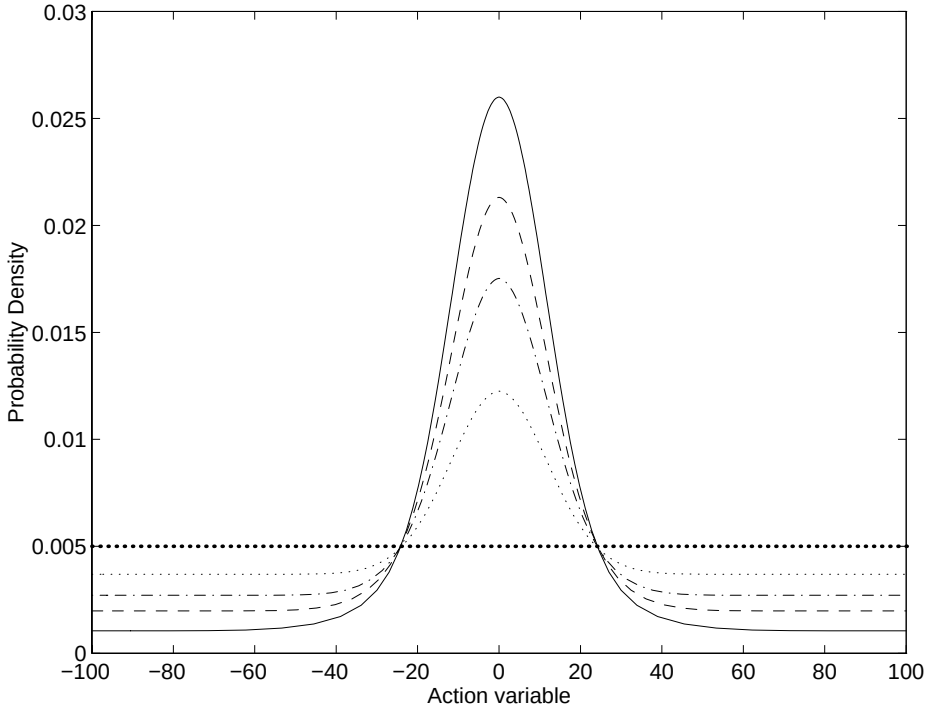

Fig 4. CARLA Convergence.

Learning can take place continuously and will therefore continue to operate, even after an optimal response has been learnt. Also the system state will not saturate at a single value through continued reinforcement.

For implementation purposes, the value of probability density function $f(x,n)$ is stored at regular intervals Δf of probability density. The action / probability density co-ordinates then approximate the probability density function with linear interpolation determining the action value at intermediate points. With this implementation the curve is better defined at high values of probability density which are the areas of most importance.

Each CARLA can be treated as a separate learning unit associated with a single continuous action variable. As previously mentioned, the practical application of the CARLA involves the multiple action interconnected structure shown in Fig. 2. This simple modular

implementation enables multiple actions to be learnt simultaneously with interactions obtained through the dynamics of the environment.

4. EXPERIMENTAL STUDY

A standard passenger vehicle fitted with continuously variable controllable semi-active dampers and wheel and body accelerometers was mounted on a servo-hydraulic four-poster road simulator. The above learning methodology was applied to the problem of improving the ride comfort of the vehicle with a performance objective of reducing the vertical body accelerations. This is a stochastic control problem for a system with a large number of degrees of freedom and significant non-linearities for example due to the complex properties of the suspension dampers and bushes. Furthermore, the excitation from the rig actuators are unknown to the learning system.

Four learning systems, each with three actions, were implemented, one for each damper. The performance objective of each of the learning systems was to minimise the mean square vertical body acceleration at the corresponding suspension mounting points. The architecture of the learning system as applied to the vehicle suspension control problem is summarised in Fig. 5.

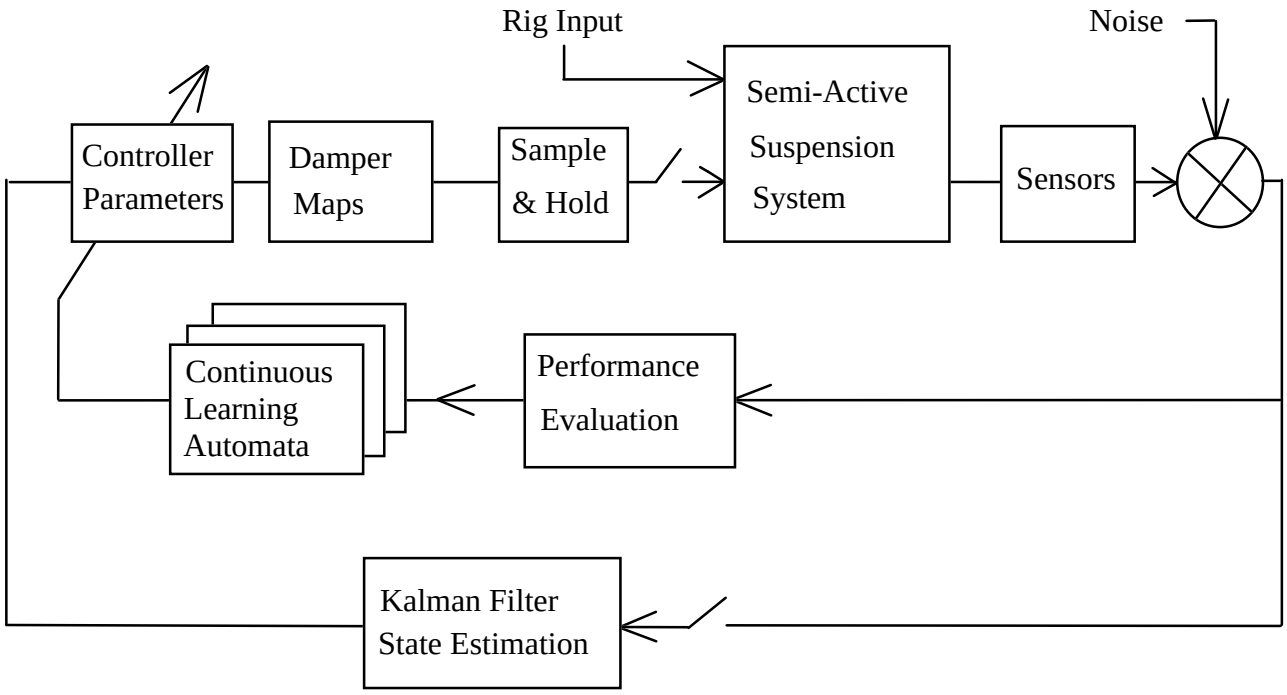

Fig. 5. System Diagram.

The following variables were used for feedback: suspension deflection (x_s), wheel velocity (v_w) and body velocity (v_b) at each corner of the vehicle. The controller then commands the desired damper force in the simple form

$$F_s = k_1 x_s + k_2 v_s + k_3 v_b \tag{11}$$

with the learning system being used to determine the controller parameters k_1, k_2, k_3. The specified ranges for these parameters have been based on the typical values reported in the literature on active and semi-active suspension control [7]. The action variable of each CARLA now corresponding to a controller parameter. This desired force is used together with the estimated velocity across the damper to determine the control valve setting of the damper. This can be achieved by using a previously determined set of non-linear maps defining damper force as a function of the velocity across the damper.

The rig excitation was created with each of the four corner being driven independently. Each rig actuator was driven according to a stationary random process with prescribed displacement spectral density function $s(f) = c \cdot f^{-2.5}$ with $c = 6 \times 10^{-6}$ and band-limited to the frequency range 0.2 to 20Hz. A VAX workstation was used to drive the four poster rig with updates to the position of the computer controlled hydraulic actuators applied at a sampling frequency of 204.8Hz. The vehicle, a medium sized saloon car, was fitted with continuously variable dampers and instrumented with sensors at each corner of the car. The desired control action converted to a voltage signal is sent to a damper driver module. This converts the signal into a pulse-width modulated square-wave current signal supplying the solenoid valve of the corresponding damper. The pulse width modulates the damping rate and the oscillating nature of this signal continuously vibrates the solenoid valve to prevent sticking and to improve the dynamic response of the actuator. The sensor set at each wheelstation comprises two piezo-resistive type accelerometers to measure wheel and body acceleration in the range $\pm 150\,\mathrm{m/s^2}$ and $\pm 20\,\mathrm{m/s^2}$ respectively. These were rigidly mounted in the vertical plane at each wheelhub, and on the vehicle body at the top pinchbolt of each damper. A cut out section in Fig. 6. shows the sensor position for the front corner of the vehicle. Single-turn rotary potentiometers, calibrated on previous rig tests [9], were also mounted at each corner to measure the suspension deflection. The sensors were connected to a signal conditioning module to amplify each of the transducer signals, and apply an analogue anti-aliasing Bessel filter. The filter induces a time delay of $500\,\mu\mathrm{s}$ at pass frequencies up to around 300Hz. In order to reduce the signal conversion hardware to a single A/D and a single D/A converter a multiplexor was employed between the signal conditioning and digital signal processor. A previously designed Kalman filter [10] was used to provide state estimates for feedback control.

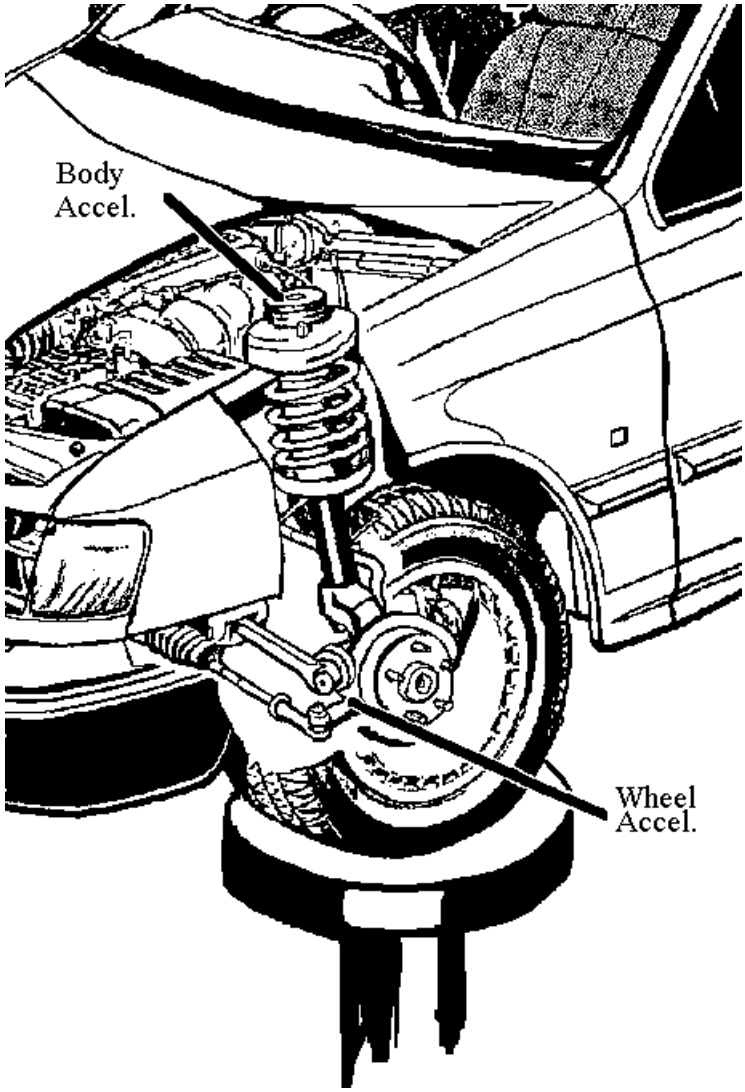

Fig. 6. Front Corner of Vehicle

The CARLA learning algorithm, implemented in 'C', was executed in real-time on a 486DX2-50. The resulting controller gains were sent to a TMS320C30 digital signal processor operating at 500 Hz, which provided the state estimation, performed the closed loop control and returned the dynamic cost. Details of how the CARLA algorithm was implemented are given below. The learning interval of 16 seconds is necessary, not for the speed of the learning system, but for the vehicle system dynamics. The interval was chosen to average out the effects of input variation and to determine the average control effect to over several cycles of low frequency body bounce acceleration.

CARLA Algorithm

1. Set iteration $n = 0$;
2. Define the action set $A(n) = \{k_1(n), k_2(n), k_3(n)\}$ such that $k_i \in [k_{i,\min}, k_{i,\max}]$ for $(i = 1,2,3)$.
3. Define $f_i(k_i, n)$ to be the probability density function for k_i at iteration n.
4. Initialise $f_i(k_i, n)$ for $i = 1,2,3$ to a uniform distribution between the defined limits.
5. Repeat
   - Using a pseudo-random number generator for each action select $z_i(n)$ for uniformly between 0 and 1. $(i = 1,2,3)$
   - Select $k_i \in A(n)$ where the area under the probability density function is
     $$\int_{k_{i,\max}}^{k_i(n)} f(x,n) = z_i(n)$$
   - Run Suspension System over a Δ second time interval.
   - Evaluate the cost or performance index J, where
     $$J(n) = \frac{1}{\Delta}\sum_{t-\Delta}^{t} \dot{v}_b^2$$
   - Append to *R* and evaluate the minimum, $J_{\min}$, and median, J_{med}, values of *R*.
   - Evaluate $\beta(n)$ via eqn (5)
   - Update the probability density functions $f_i(k_i, n)$ using eqn (6)
   - Increment iteration number n.

Fig. 7. CARLA Implementation

Parameter Values

Δ = 16 seconds

Δf = 0.01

m = 500 (values in the reference set *R*)

g_w = 0.02

g_h = 0.3

$k_{1,\min}$	=	0	$k_{1,\max}$	=	20000
$k_{2,\min}$	=	0	$k_{2,\max}$	=	2000
$k_{3,\min}$	=	-6000	$k_{3,\max}$	=	0

5. CARLA CONTROLLER PERFORMANCE

Three independent examples of the learning system were run on the rig with each example using a different random driving input. The first two tests were each run for 2900 iterations, which represents about 13 hours of running time; the third test was extended to 3300 iterations. The resulting controllers, denoted LC 1-3, were then tested on an independent 60 second sample of rig excitation for evaluation purposes. In Table 1 the r.m.s. body accelerations are given as percentage reductions from a best-case constant setting (reference passive control). Firm and Soft passive damper settings were also tested and are included in the Table.

Percentage Increase	Corner A	Corner B	Corner C	Corner D
Firm	49.17	39.83	46.63	32.86
Soft	-2.077	5.171	10.49	0.407
LC 1	-6.450	-5.635	-3.240	-5.985
LC 2	-5.972	-5.648	-0.793	-7.008
LC 3	-10.08	-9.413	-8.706	-13.80

Table 1. Percentage reduction in r.m.s. body acceleration. [comparison with a reference passive damper setting. Negative values denote increased body acceleration].

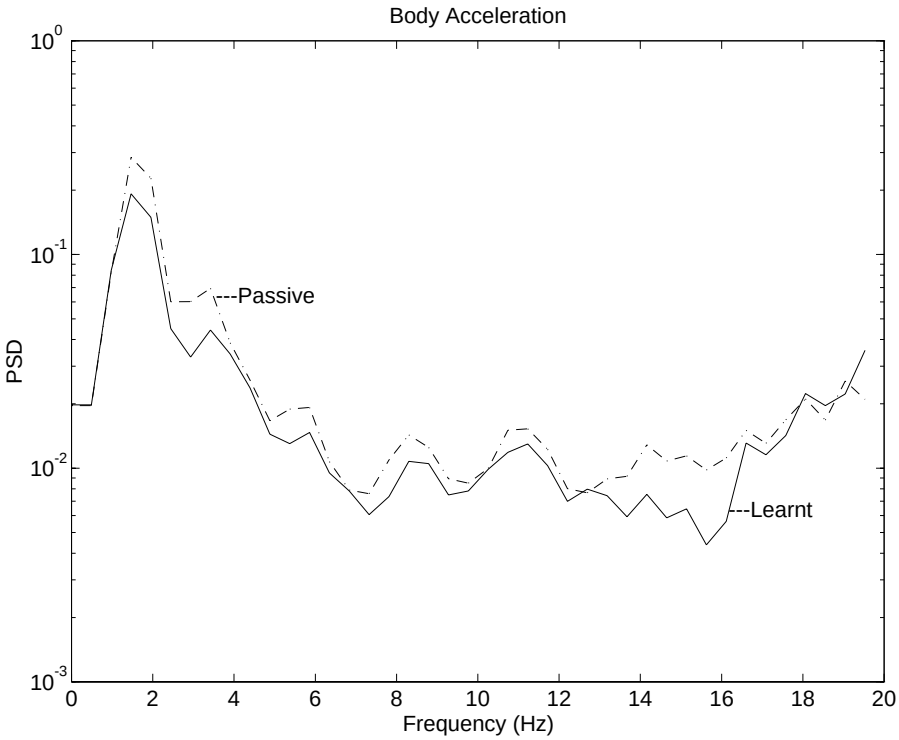

Fig. 8. A comparison of the power spectral density function for the best passive damper setting and a learnt controller

The results indicate that the semi-active system can achieve improvements of the order of 10%, though the extra learning time for controller LC 3 may suggest that the system was still in a process of improving its performance and longer runs may be even more beneficial.

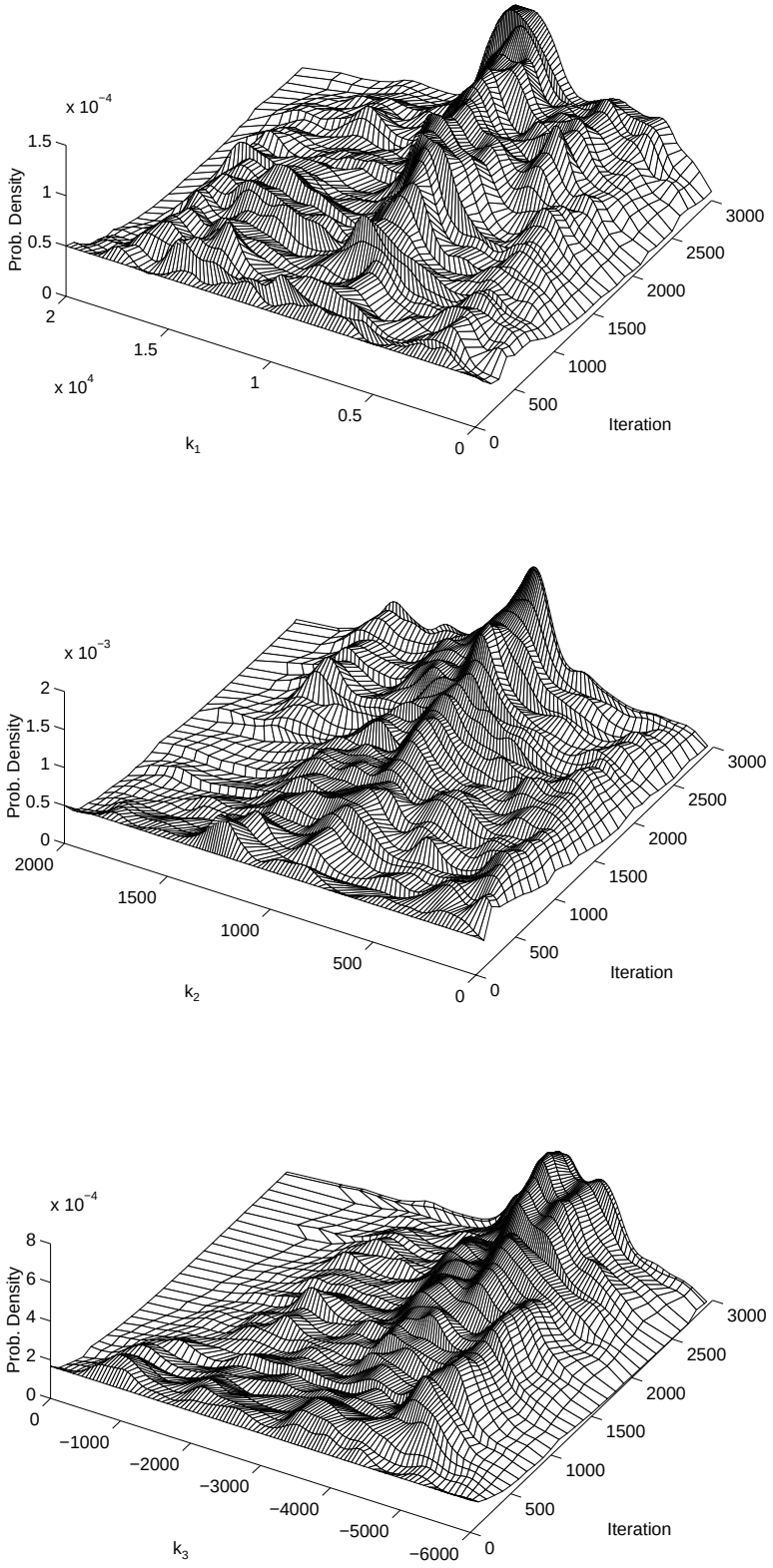

Fig. 9. The variation in probability density for each controller parameter.

Fig. 8 shows the power spectral density function of a typical learnt controller and the best passive damper setting. This indicates improvements can occur across a wide frequency range. Fig. 9 shows how the probability density function varies over each iteration for the controller parameters of one corner in a typical test. There is a clear trend of convergence as the three probability functions become increasingly sharply peaked. The locations of the peak values at the final iteration shown here were used for the parameters in controller LC1; a similar though independent process leads to each of the LC2 and LC3 parameter sets. The mean reduction in cost (averaged over 100 iterations) is shown in Fig. 10. The mean cost has not completely levelled off indicating that slight further improvements in performance may still be possible.

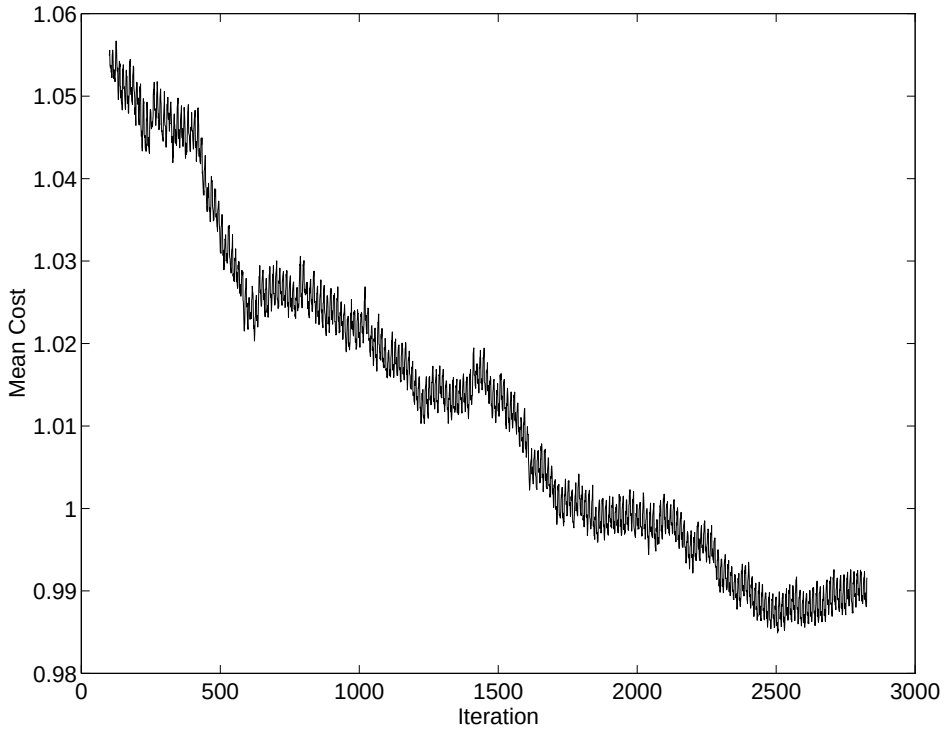

Fig. 10. Reduction in mean cost per iteration

6. DISCUSSION AND CONCLUSIONS

The Continuous Action Reinforcement Learning Automata have been shown to operate successfully in a highly stochastic environment. The learning methodology was capable of improving the system performance despite high levels of noise on the rig input and sensors. With each of the dampers being controlled separately, interactions occur via the vehicle's body structure. These effects contributed to the relatively slow learning time taken, despite the simplistic structure of the controller. This initial study has, however, demonstrated that the methodology is practicable, and has the potential for use as a general tool for industrial control system design.

The CARLA benefits from having relatively few free parameters, the most significant being the height and width of the Gaussian neighbourhood functions. In fact the learning is not overly sensitive to these parameters, although the speed of learning is affected to a small extent. The structure of the algorithm is relatively simple, with one CARLA per variable to be learnt. These can be easily combined to create general configurations, the interconnections being through the dynamics of the environment. The CARLA may share the same cost function with one overall goal or they may each use separate cost functions, to achieve there own individual sub-goals. The simple and general structure of the algorithm makes the technique particularly suited to distributed control applications and implementation in parallel processing hardware.

Overall convergence rate is determined by (i) iteration time, which is constrained by the system dynamics, and (ii) CARLA convergence rate, which is a function of the CARLA parameters g_w, g_h as well as the environmental noise levels. It is thought that in this study area the best opportunity for improving convergence rates lies with reducing the duration of each iteration from the current 16 seconds. This is far from trivial however, and requires a

more detailed assessment of the effects in the vehicle of the different control actions; this issue is currently being studied by the authors.

The learning system, as with the learning automata methodology on which it is based, can be used for on-line learning in stochastic environments and because of the randomised nature of the action selection should have the potential to avoid problems caused by local minima.

More conventional controller design for such a system would have incurred very significant overheads in the development and implementation time. Initially some form of systems modelling would be required. The system parameter would need to be identified and these validated on the vehicle before simulation studies were conducted. The final controller design would then have to be separately verified on the real system hardware. The control system developed in this study required no modelling - the system learns on the real hardware. This feature should enable the methodology to be used for fast controller design and system development in other areas.

The technique has been shown to be successful on a rig based system with a significant improvement over a reference passive setting being achieved. It should be possible to extend these techniques to allow road based learning in the near future. The complexity of the controller structure will also be increased and methods of reducing the learning time that is required will be also need to be addressed. The operation of the continuous learning automata to perform the state estimation functions will also be examined to eliminate the need for a pre-designed Kalman filter.

ACKNOWLEDGEMENTS

This research is supported by the Engineering and Physical Science Research Council (EPSRC) grant no. GR/J82652 and is in collaboration with the Ford Motor Company.

REFERENCES

1. Karnopp D., Design Principles for Vibration Control Systems Using Semi-Active Dampers. *Trans. of the. ASME,* Vol. 112, 448-455 (1990)

2. Boyan J., Modular Neural Nets for Reinforcement Learning of Game Strategies, *Master's thesis*, Cambridge (1992).

3. Lin L.J., Programming robots using reinforcement learning and teaching, *Proc. 9th National Conf. on Artificial Intelligence* 781-786 (1991).

4. Wu Q,H., Learning Co-ordinated Control of Synchronous Machines in Multi-machine Power Systems, *Proc. IEEE Systems, Man and Cybernetics 1993 Conference*, Vol. 3, 728-733 (1993).

5. Thrun, S., and Schwartz, A., Issues in Using Function Approximation for Reinforcement Learning, *Proc. 4th Connectionist Models Summer School*. (1993).

6. Frost, G.P., Gordon, T.J., and Wu, Q.H., The Application of Learning Automata to Advanced Vehicle Suspension, *Control of Vibration*, 5-8 Sept. Bath (1994).

7. Gordon, T.J., Marsh, C., and Wu, Q.H., Stochastic Optimal Control of Active Vehicle Suspensions Using Learning Automata, *Proc. I.Mech.E. Part I*, Vol. 207, 143-152 (1993).

8. Narendra, K.S., and Thathachar. M.A.L., *Learning Automata: An Introduction*, Prentice Hall, London. (1989).

9. Best, M.C., On The Modelling Requirements For The Practical Implementation of Advanced Vehicle Suspension Control, Doctoral Thesis, Loughborough University, U.K., (1995).

10. Best, M.C., Gordon T.J. Crawford I.L, Hand P., Real Time Estimation of Vehicle Filtering. *Proc. FISITA Congress*, China, Vol.2, 16-24 (1994).