Journal of Vibration and Control
XX(X):1–21

DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Deep Learning Framework for Controlling an Active Suspension System

Aleksey Konoiko[1], Allan Kadhem[1], Islam Saiful[1], Navid Ghorbanian[1], Yahya Zweiri[1,2] and M. Necip Sahinkaya [1]

Abstract

In this paper, a feed-forward deep neural network and automated search method for optimum network structure are developed to control an active suspension system. The network was trained through supervised learning using backpropagation algorithm. The training data was generated from an optimal PID controller tuned based on a full state feedback optimal controller. The trained network was implemented in an active suspension system test rig for a quarter car model and was initially tested in simulation under parameter uncertainties. Experimental results showed that the developed deep neural network controller outperforms the optimal controller under uncertainties in terms of reducing the sprung mass acceleration and actuator energy consumption, with a 4% and 14% reduction respectively.

Keywords

Active Suspension System, Control Systems, Control Design, Deep Learning, Deep Neural Networks, Machine Learning, Backpropagation, Optimal Control, Optimal PID Controller

Introduction

The suspension system is a fundamental vehicle component that aims to ensure passengers comfort by suppressing oscillations caused by road irregularities and to

[1] Faculty of Science, Engineering and Computing, Kingston University London, London SW15 3DW, UK
[2] Khalifa University of Science and Technology, P.O. Box 127788, Abu Dhabi, United Arab Emirates

Corresponding author:
M. Necip Sahinkaya, Faculty of Science, Engineering and Computing, Kingston University London, London SW15 3DW, UK.
Email: M.Sahinkaya@kingston.ac.uk

Prepared using sagej.cls *[Version: 2017/01/17 v1.20]*

improve handling by keeping constant ground contact under any condition to maximise traction Aly and Salem (2013). The development of suspension technology attempts to transition traditional passive suspension systems to active suspension systems in more efficient and cost effective ways Alexandru and Alexandru (2011). In an Active Suspension Systems (ASS), an actuator is used in combination with springs and dampers to exert a force that controls suspensions dynamic behaviour in order to adapt to road profile and achieve higher level of performance Fischer and Isermann (2004). Because of non-linearities and time-varying characteristics of suspension systems, they are difficult to model for ASS control systems. Therefore, major research focused on ASS controller design is required.

Simple solutions were attempted initially in order for controllers to be able to deal with uncertainties such as the use a self-tuning pole assignment and fuzzy logic to adjust the gains of the controller Ramsbottom et al. (1999). Another common example of control systems based solution was the Sliding Mode, a form of non-linear adaptive control scheme which allows dealing with suspension non-linearities due to changes in stiffness and dampening characteristics. It uses a sky-hook damper system model without the need for real time measurement of the road profile Kim and Ro (1998). With advances in the technology, more complex controllers were developed. A programmable full state feedback controller based on linear quadratic control and pole placement techniques for a linearised model was developed however could not achieve satisfactory sprung acceleration suppression Watton et al. (2004). There are examples of automation of controller design that uses a modelling-free approach based on adaptive sliding controller where a self-tuning fuzzy scheme is used to compensate for modelling uncertainty and improve vibration suppression Chen and Huang (2005). An interesting control strategy uses an inertial delay linear quadratic Gaussian observer which estimates states, uncertainties and disturbances of a system simultaneously by using only the sprung mass position sensor was developed. Gupta et al. (2016) designed a multi-level controller formed by two controllers. The upper supervisory controller computes the damping force with linear quadratic control theory and selects the passive or active mode for the suspension while the lower level controller feeds the actuator signal compensating for dynamic friction torque. Two estimators are used for system states while an adaptive observer is used for damper friction and actuator friction Shin et al. (2016). Along with control systems based design, solutions were developed through algorithms from Computer Science such as Genetic algorithms, that were used to find optimum control parameters for active suspension system in a half car model using minimum suspension travel as the objective of the algorithm Tsao and Chen (2001).

Neural networks (NNs) were used in combination with more conventional controllers in order to aid or enhance the controller performance. Huang et al. proposed a combination of adaptive control based on sliding mode and NN based on radial basis function to deal with the non-linear and time-varying characteristics of a hydraulic ASS. The NN tunes its weights online and learns from the sliding mode controller to suppress oscillations due to road irregularities Huang and Lin (2007). A state-observer-based feedback control is developed using an adaptive radial basis function neural network (RBFNN) to deal with system non-linearities and avoid actuator saturation while another

NN estimates the states. The parameters of the controller are optimised through particle swarm optimisation scheme Zhao et al. (2016). Metered et al. (2010) developed a recurrent neural network (RNN) controller based on inverse dynamics derived from the Bouc-Wen model of a semi active suspension system with magnetic damper actuator. Hardware-in-the-loop (HIL) simulations with different road disturbances showed that the RNN offered improved actuator voltage signal which ensures extended service life and lower power requirement. Heidari and Homaei (2013) have designed a PID controller for active suspension system which its gain parameters are obtained using NN trained with Levernberg-Marquardt algorithm. An NN is used to switch pre-programmed control laws, which focus on either ride comfort, ride safety or a trade-off of these criteria, based on the system states. The NN then optimises the parameters of the control schemes based on the input it receives Dessort and Chucholowski (2017). An adaptive neuro-fuzzy controller for an active suspension system was developed using an LQR approach for training. The authors removed the need for the dependency of a stochastic observer from the system and used Kalman filter to determine the system states Rashid et al. (2016). Other approaches developed NNs for ASS to carry out other tasks, for example, in the work of Qin et al. (2018), a novel road roughness detection and classification method was proposed where the system response in terms of rattle space (relative unsprung mass displacement), unsprung mass and sprung mass accelerations are used as inputs and the road roughness classes as the outputs. On-board camera stabilisation is another important application for autonomous cars which relies heavily on computer vision, so an ASS controller was developed using RBFNN in combination with LQR full-state feedback control Zhao et al. (2015). The road disturbance is sensed in advance by either optical sensor or measurement of vehicle dynamic response then RBFNN improves the accuracy of the controller in real-time.

DNN is rarely used on its own in the control of active suspension systems as proposed here. Previous neural network models were combined with Neuro-fuzzy, Sliding Mode, LQR, PID or similar to tune their parameters either offline or in real time. The research on RNN model reported in Metered et al. (2010) is the closest to our case but it was used for a semi-active suspension system utilising a magnetic damper which is significantly different than the Active Suspension System. The contribution of this paper consists of complete substitution of the conventional controller with a Deep Neural Network (DNN) controller. Also, a script was developed to derive the optimal structure for the DNN which automatically iterates over potential structures. The optimum DNN controller is implemented in a Hardware in the Loop (HIL) system and its performance is validated in real time with real road data. The DNN controller demonstrated a better performance in terms of energy consumption and riding comfort than an optimal PID controller.

Methodology

Active Suspension System Modelling

The quarter-car ASS model consists of two masses, each supported by a spring and a damper as shown in Figure 1. The *sprung mass*, m_s, represents the mass of the car body

while the *unsprung mass*, m_{us}, represents the wheel. The spring k_s and the damper b_s support the body weight over the wheel. The spring k_{us} and the damper b_{us} model the stiffness and the damping of the tire in contact with the road. The actuator is positioned alongside the spring and the damper and it is represented by the force F_A.

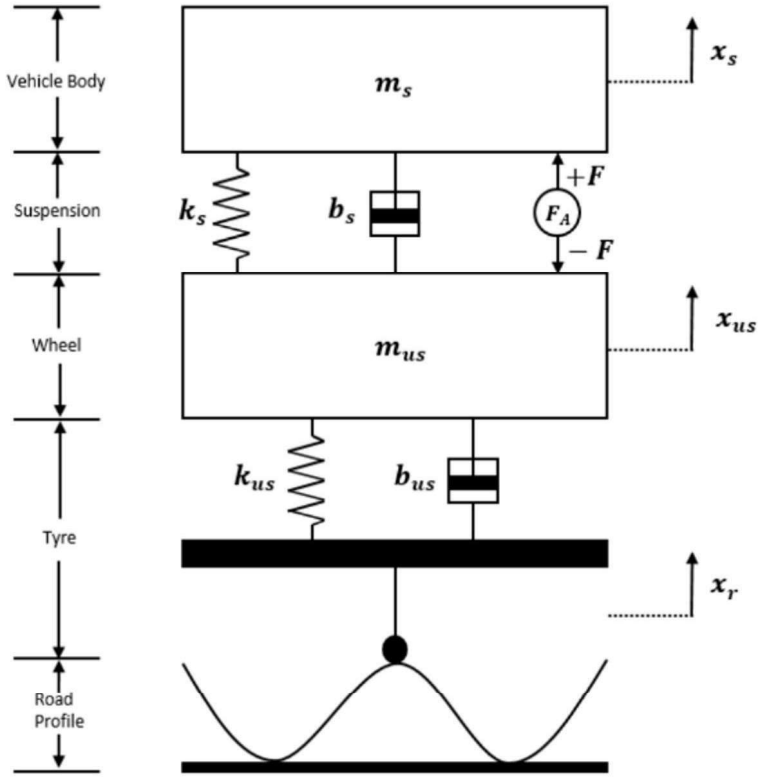

Figure 1. Double Mass-Spring-Damper used to model Active Suspension

The state space approach is used to model the quarter-car system which is represented as a multi-input multi-output linear system by the following state space form:

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\,\mathbf{x} \;+\; \mathbf{B}_1\,\mathbf{u} + \mathbf{B}_2\,\mathbf{f} \\ \mathbf{y} &= \mathbf{C}\,\mathbf{x} + \mathbf{D}\,\mathbf{u} + \mathbf{D}'\,\mathbf{f} \end{aligned} \tag{1}$$

where $\mathbf{x} \in \Re^{N\times 1}$ is the vector of state variables, $\mathbf{u} \in \Re^{M\times 1}$ is the vector of control inputs, and $\mathbf{f} \in \Re^{K\times 1}$ is the vector of external inputs and disturbances, and $\mathbf{y} \in \Re^{P\times 1}$ is the vector of outputs.

$$\mathbf{x} = \begin{bmatrix} x_s - x_{us} \\ \dot{x}_s \\ x_{us} - x_r \\ \dot{x}_{us} \end{bmatrix} \tag{2}$$

$$\mathbf{u} = \begin{bmatrix} F_A \end{bmatrix}, \qquad \mathbf{f} = \begin{bmatrix} \dot{x}_r \end{bmatrix} \tag{3}$$

$$\mathbf{y} = \begin{bmatrix} x_s - x_{us} \\ \ddot{x}_s \end{bmatrix} \tag{4}$$

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & -1 \\ -\frac{k_s}{m_s} & -\frac{b_s}{m_s} & 0 & \frac{b_s}{m_s} \\ 0 & 0 & 0 & 1 \\ \frac{k_s}{m_{us}} & \frac{b_s}{m_{us}} & -\frac{k_{us}}{m_{us}} & -\frac{b_s+b_{us}}{m_{us}} \end{bmatrix} \tag{5}$$

$$\mathbf{B}_1 = \begin{bmatrix} 0 \\ \frac{1}{m_s} \\ 0 \\ -\frac{1}{M_{us}} \end{bmatrix}, \qquad \mathbf{B}_2 = \begin{bmatrix} 0 \\ 0 \\ -1 \\ \frac{b_{us}}{m_{us}} \end{bmatrix} \tag{6}$$

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \frac{-k_s}{ms} & \frac{-b_s}{m_s} & 0 & \frac{b_s}{m_s} \end{bmatrix} \tag{7}$$

$$\mathbf{D} = \begin{bmatrix} 0 \\ \frac{1}{m_s} \end{bmatrix}, \qquad \mathbf{D}' = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{8}$$

The states can be defined such that they reflect the system performance parameters, given in equation (2). The first state represents the suspension deflection/travel. The second state is the vehicle body (sprung mass) vertical velocity. The third state is the wheel deflection which is a measure of road handling. The fourth state is the wheel vertical velocity. The control input is the actuator action and the external input to the system is the rate of change of road surface, equation (3). The first measured output of the system is the suspension travel, and the second measured output of the system is the vehicle body acceleration, as given in equation (4). The parameter values of the active suspension system are given in Table 1.

Table 1. Suspension System Parameters

Symbol	Name	Value
m_s	Sprung Mass	2.45 kg
m_{us}	Unsprung Mass	1 kg
k_s	Suspension Stiffness	900 N/m
k_{us}	Tire Stiffness	2500 N/m
b_s	Suspension Damping Coefficient	7.5 Ns/m
b_{us}	Tire Damping Coefficient	5 Ns/m

Passive Suspension System: Passive suspension system is represented by the open loop response of the systems which can be obtained by setting $\mathbf{u} = 0$ in equation (1). This corresponds to no control action from the actuator. The open loop response of the sprung mass travel is shown in Figure 2. A square signal with amplitude of 0.02 m and period of 3 s is used to simulate the road profile x_r as worst case scenario. The result illustrates that the maximum overshoot is almost 100% for the given road profile which causes severe vibrations that negatively affect passenger comfort and driving handling.

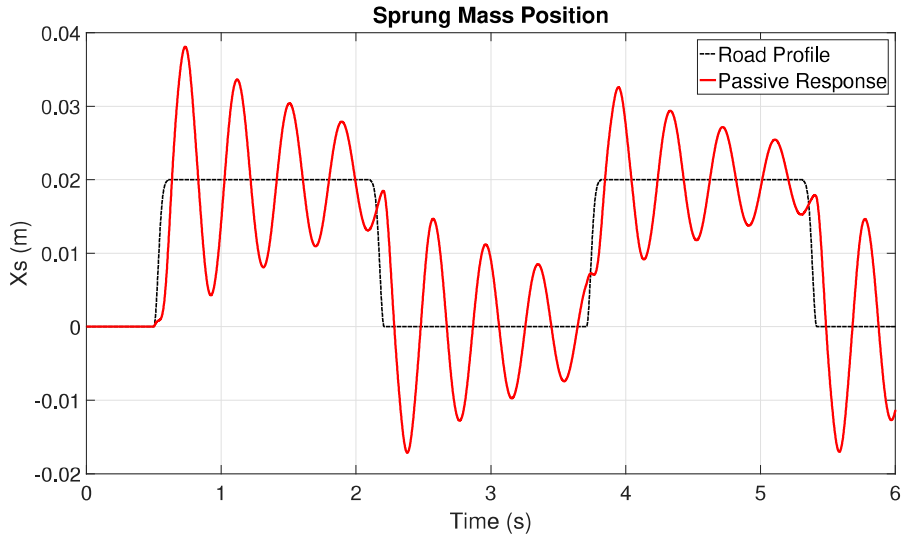

Figure 2. Simulation Open loop response

Active Suspension System: To turn the system from passive to active suspension, a controller was tuned with the following aims: no overshoot of the car body with reference to the road profile; reduction in the vertical acceleration of the car body; maintaining contact between the wheels and the road; ability to function across different road profiles and low actuator energy consumption. With these aims, an optimal Proportional Integral Derivative (PID) controller was tuned using LQR method highlighted in Alkhoori et al.

(2017); Bin Safwan et al. (2018). The input and output for the controller are velocity and force respectively which were used as the training data for the deep neural network controller. The optimum PID gain values used are $K_P = 89.9882$ Ns/m and $K_I = K_D = 0$ (i.e. velocity feedback only).

Training of Deep Neural Networks

Training Data: Training data was generated from simulations of the optimal PID controller using the closed loop response to square wave with different amplitudes and sinusoidal road profiles with a sampling time of 1 ms. Vehicle body velocity is used as input data to the DNN while actuator force signal is used as target data. The velocity and the corresponding force signal data points for each millisecond are treated as training sets (batches). This training data aims to train the DNN to model the behaviour of the optimal PID controller and to be able to predict the correct actuator force signal to minimise vehicle body oscillations for unknown road profiles. The proposed method is summarised in Figure (3) :

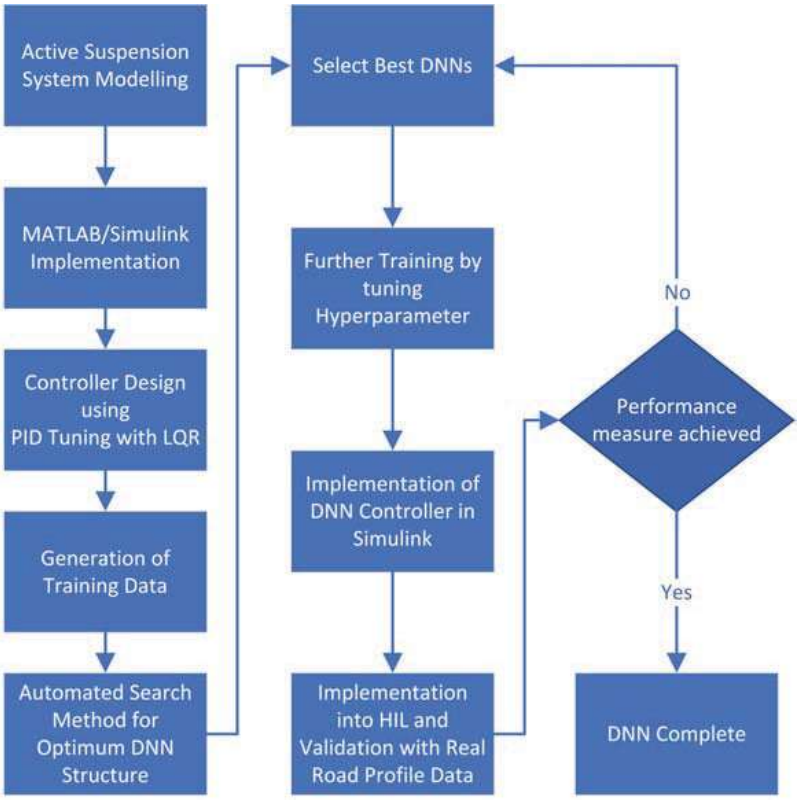

Figure 3. Training Procedure Workflow

The training data has to be mapped into an appropriate range. The input and output data are mapped based on the activation functions (AF), Table (2), at the first hidden layer and at the output layer, respectively. The output of the network is reverse mapped to the original range to generate the force control signal. Mapping must also consider the physical limits of the real system. Therefore it is mandatory to make sure that the training data is within safe margins of the given system, in this case, force signal is limited to 20N.

Table 2. Activation Functions

Function	Equation	Derivative
$Sigmoid$	$\frac{1}{1+e^{-}x}$	$f'(x) = x(1-x)$
$Tanh$	$\frac{2}{1+e^{-}2x} - 1$	$f'(x) = (1-x)(1+x)$
$ReLU$	$\begin{cases} x & \text{for } x \geq 0 \\ 0 & \text{for } x < 0 \end{cases}$	$\begin{cases} 1 & \text{for } x \geq 0 \\ 0 & \text{for } x < 0 \end{cases}$

Each AF presents inherent disadvantages, for example the Sigmoid function becomes saturated at its range ends, (0 and 1) and so does hyperbolic tangent (1 and -1), which can lead to vanishing gradient. Practical experience with the data showed poor results or unsuccessful training when using the same activation function in every layer, even when using ReLU. However, the combination of all three was more effective for our problem.

The training procedure has three stages: training, testing and validation. In general, the training and testing data are generated from random portions of the same data set by assigning percentages, i.e. 70% for training and 30% for testing, while validation requires a data set that the DNN had never seen before. Data using a real road profile was used for validation purpose.

Weights Initialisation: Initial weights are generated with random numbers to have a starting point. For the weight matrix of a layer, the number of columns is equal to the number of inputs into the layer while the number of rows corresponds to the number of hidden nodes or outputs from the layer. In a feed-forward DNN, the outputs from one layer are the inputs to the next layer.

Deep Learning: Backpropagation (Backprop) with Stochastic Gradient Descent (SGD) is the main Deep Learning (DL) algorithm used for supervised learning of DNNs. In each training cycle (epoch), the inputs propagate within the network, which is known as forward-pass, and an output is generated. Then, during Backprop, the error between the output and target data is calculated using a Mean Squared Error (MSE) or Cross Entropy (CE) loss functions, and propagated backwards as shown in Figure 4. From output to input, SGD computes the delta (δ) which defined as the derivative of the activation function multiplied by the error then generating weights updates ΔW at each layer in

order to minimise the loss function value. The pseudocode of a DNN with 3 hidden layers is given in Algorithm (1).

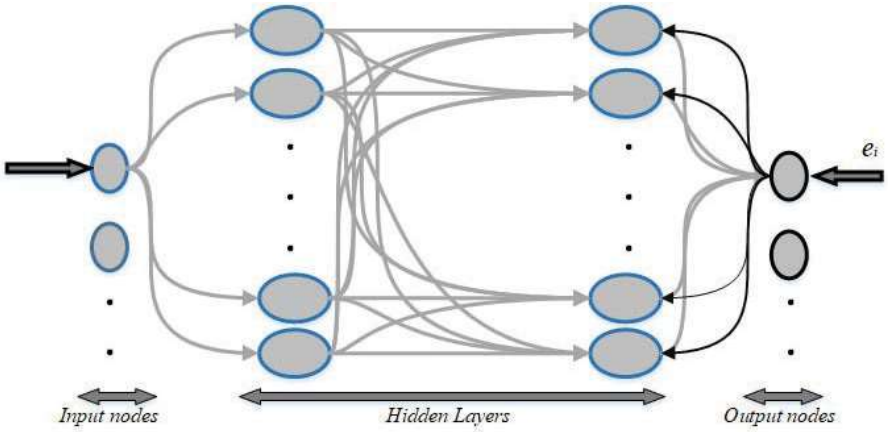

Figure 4. Back-propagation

Implementation

Automated Search of Optimal Structure

There are a number of parameters that can affect the training performance such as:

- Type and order of Activation Functions
- Nodes per Layer
- Number of Hidden Layers
- Epochs
- Learning Rate
- Momentum

To this end, a program was developed to automate the process of varying each parameter in order to assess their effect on the performance of the DNN model. The number of starting layers, starting nodes and starting epochs along with their respective range and step size are specified in the program. The program then tests these parameters for all permutation of activation functions. The DNN iteration script was created in multiple versions where each version cycled through a limited set of structure combinations. Created scripts were executed in multiple computers with Intel Quad Core i7-6700 CPU with clock frequency of 4Ghz and 32GB of RAM. This allowed to shorten the total time to complete all the iterations. The output data from all the runs were collected in MATLAB to identify the best performing structures. Determining optimum DNN structure normally involves a trial-error method, without knowing whether an optimum has been achieved. The proposed automated search method ensures that we

Algorithm 1 Training Algorithm Function

function $[W1, W2, W3]$ = TRAINING($W1$, $W2$, $W3$, $inputs$, $targets$, $batches$, $learning\_rate$, $momentum$, $AF1$, $AF2$, $AF3$, $epochs$)

$mmt1$=zeros(size($W1$)); ▷ Momentum Initialisation
$mmt2$=zeros(size($W2$));
$mmt3$=zeros(size($W3$));

for $batch = 1 : batches$ **do**

$x = inputs(batch, :)'$; ▷ Feed-forward

$v1 = W1 * x$;
$y1 = AF1(v1)$;
$v2 = W2 * y1$;
$y2 = AF2(v2)$;
$v3 = W3 * y2$;
$y3 = AF3(v3)$;

$d = Target(Batch, :)'$;

$error = d - y3$; ▷ Backpropagation
$delta3 = error$;
$e2 = W3' * delta3$;
$delta2 = y2. * (1 - y2). * e2$;
$e1 = W2' * delta2$;
$delta1 = y1. * (1 - y1). * e1$;

▷ Weights Update
$dW3 = learning\_rate * delta3 * y2'$
$mmt3 = dW3 + momentum * mmt3$;
$W3 = W3 + mmt3$;

$dW2 = learning\_rate * delta2 * y1'$
$mmt2 = dW2 + momentum * mmt2$;
$W2 = W2 + mmt2$;

$dW1 = learning\_rate * delta1 * x'$;
$mmt1 = dW1 + momentum * mmt1$;
$W1 = W1 + mmt1$;

have the optimum structure before using the network. It requires computing power, but

it is done offline and only once to determine the best structure for the application. When the DNN controller is implemented in real time it is fast and efficient.

It was found that the DNNs performed better as the number of layers was increased however, increasing the number of layers beyond 6 would drastically increase the total computational time to beyond two weeks without significant improvement in the accuracy. For this reason 6 layers were chosen as it provided acceptable solutions with reasonable computational time within 2-3 days. Eighteen nodes per layer produced the best results; using a higher or lower value than 18 nodes caused the output to overfit or underfit. The total number of permutations for a DNN consisting of 6 layers and 3 types of AF is 729 and this becomes the number of iterations for the DNN.

Optimal Structure

The structure of the feed-forward DNN model in iteration 147 from Table (3) and illustrated in Figure (5) performed better than any other structure having lower steady state error and peak error, however the accuracy was not sufficient for training and testing.

Table 3. Best Performing Structures with 6 Layers and 18 Nodes per Layer

Iteration	**Hidden Layer (HL) Number**					**Output**	**Peak**
No.	**HL1**	**HL2**	**HL3**	**HL4**	**HL5**	**Layer**	**MSE**
147	Sig	ReLU	Tanh	ReLU	Sig	ReLU	3.99E-4
150	Sig	ReLU	Tanh	ReLU	ReLU	ReLU	2.64E-3
201	Sig	Tanh	ReLU	ReLU	Sig	ReLU	4.4E-3
347	ReLU	ReLU	Sig	Tanh	ReLU	Sig	5.9E-3
419	ReLU	Tanh	Sig	ReLU	ReLU	Sig	4.8E-3

Variable Learning Rate: In order to tackle the accuracy issue a practical approach was employed in which the DNN is trained using a learning rate of 0.01 for 100 epochs to achieve rapid convergence at the beginning and then trained further with a learning rate of 0.0001 in steps of 5 epochs until the accuracy became satisfactory and independent of number of epochs. The value used for momentum is 0.9 which is selected after using variety of values and found to provide good stability and convergence. The final result is illustrated in Figure (6).

DNN Controller into Simulink

To test the output of the DNN in simulated environment, a Simulink model of the system is created, of which a diagram is shown in Figure (7).

The DNN is represented by the Deep Learning controller which consists in a code reproducing the structure of iteration 147, Algorithm (2). The trained weights from iteration 147 are loaded into the controller and the mapped velocity input propagates through the controller to generate the actuator force signal, ReLU is used for the optimal structure output layer which is then mapped to the safe actuator range.

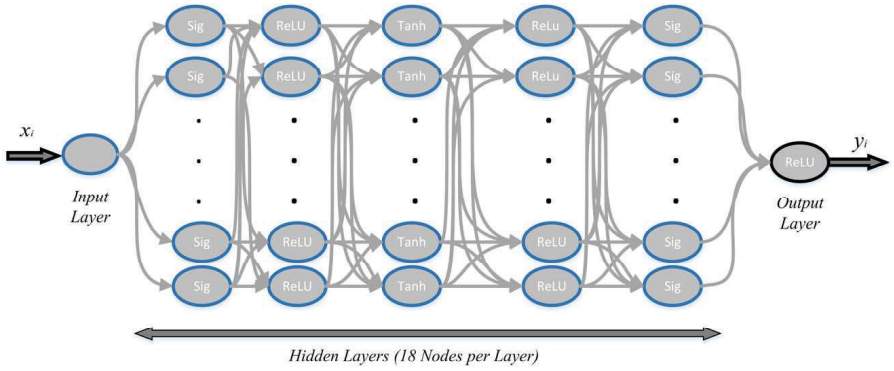

Figure 5. Optimal DNN Structure

Algorithm 2 DL Controller Function Block

function $force$ = DLCONTROLLER($velocity$, $W1$, $W2$, $W3$, $W4$, $W5$, $W6$)

$v1 = W1 * velocity$;
$y1$ = Sigmoid($v1$);

$v2 = W2 * y1$;
$y2$ = ReLU($v2$);

$v3 = W3 * y2$;
$y3$ = Tanh($v3$);

$v4 = W4 * y3$;
$y4$ = ReLU($v4$);

$v5 = W5 * y4$;
$y5$ = Sigmoid($v5$);

$v6 = W6 * y5$;
$force$ = ReLU($v6$);

Simulation Results

Simulations were performed to establish DL controller performance under (a) ideal conditions with no noise in the system and (b) parameters uncertainties.

(a) Under ideal conditions

Figure (8) compares sprung mass (vehicle body) and un-sprung mass (wheel and suspension assembly) relative positions using the DNN Controller against the optimal PID controller with respect to the road profile which is set as square waveform in order to simulate the worst case scenario.

It is found that the performance is practically identical, with an average error of 0.9% and 0.07% with respect to the sprung and un-sprung mass relative positions respectively. The DNN controller demonstrates to be able to suppress oscillations as much as the optimal controller.

Figure (9) shows the actuator force and sprung mass acceleration. The actuator has to apply the force in the opposite direction of sprung mass motion in order to suppress oscillations, for example, force undershoots at the start of the road profile irregularity, since the sprung mass tends to move upwards. The maximum performance difference found to be around 2.8%. Similarly, the sprung mass vertical acceleration results are consistent with the trend, where overshoots at the start of the road irregularity, and it displays a 3.2% difference between the DNN and the optimal PID controller.

(b) Under parameters uncertainties

The system model parameters were arbitrarily modified: the sprung mass was increased to 3.5kg, suspension stiffness decreased to 890 N/m and damper coefficients decreased to 6.5 sec. The modified system model was used to perform simulations to test the flexibility of the DNN, this means to check the ”fitting of the model” using different conditions. The results were similar to those presented in Figures (8) and (9) with negligible differences, which demonstrates successful model fitting.

Experimental Results

Testing Rig Setup

To validate the proposed DNN based controller, a Hardware-In-the-Loop (HIL) active suspension test rig was used. Diagram of the active suspension test rig is shown in (Figure 10). The active test rig emulates a quarter car model. In the experiment, position of the sprung mass, the suspension travel and the wheel travel are measured using three quadrature optical encoders with resolution 1024, 1000 and 2048 lines per revolution, respectively. The corresponding velocity signals are estimated using robust second-order high pass filters. A dual-axis accelerometer is used to measure the acceleration of the vehicle body (sprung mass) along the vertical direction. The active suspension test rig parameters are given in Table 1.

Experimental Results with Real Road Data

The DNN controller performance was subjected to real road profile on the experimental rig. The real road data that was used for validation has been retrieved from a data log of sensors integrated on a car owned by Kingston University.

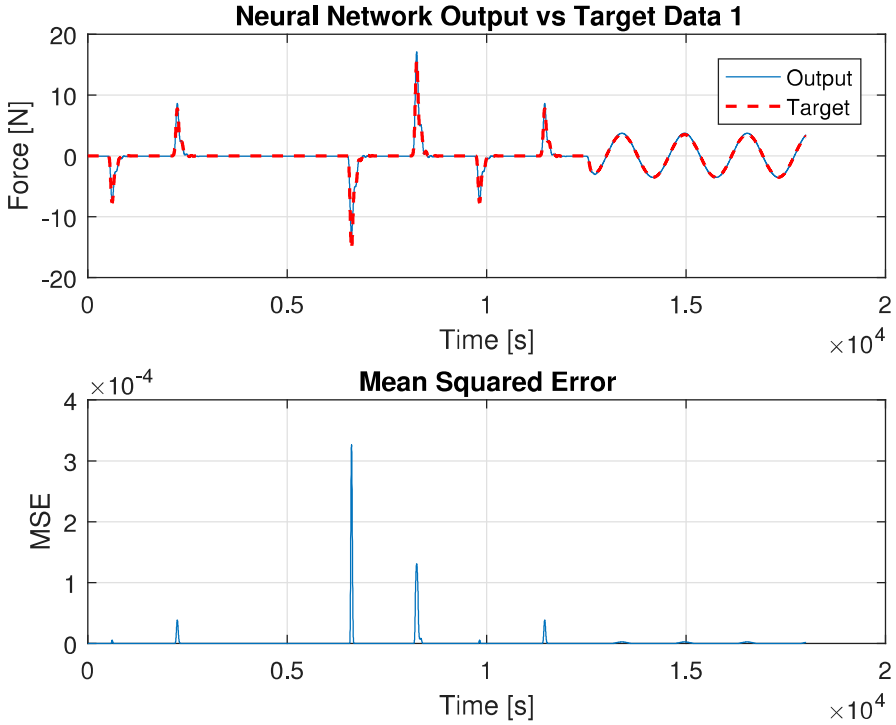

Figure 6. Iteration 147: DNN output & target vs time

Sprung and Un-sprung Mass Position vs Time: The developed DNN demonstrated good performance in terms of sprung mass displacement as shown in Figure 11, which means that amplitude of sprung mass oscillation was successfully reduced in comparison to the passive system resulting in increased comfort. The sprung mass performance of DNN is similar to the sprung mass performance of the optimal PID controller however the DNN outperforms the optimal PID controller as the un-sprung mass tracks the reference road profile better. This can be see in Figure 11 where DNN, PID and passive systems have a small millisecond delay behind the road profile which is believe to be caused by the delay in the sensors. However when ignoring this delay it is clear that the DNN controller is able to track the road profile more closely.

Actuator Force and Sprung Mass Acceleration vs Time: The real road profile has rapidly changing oscillations which make difficult to determine the performance of DNN relative to the optimal PID controller Figure 12. The mean absolute values were calculated for DNN and the PID controllers to assess their overall performance. From these values, it can be established that the DNN uses $0.6343N$ on average compared to $0.7384N$ for the optimal PID controller. This shows 14.1% lower energy consumption for the DNN controller. This energy saving is significant and very important to ensuring

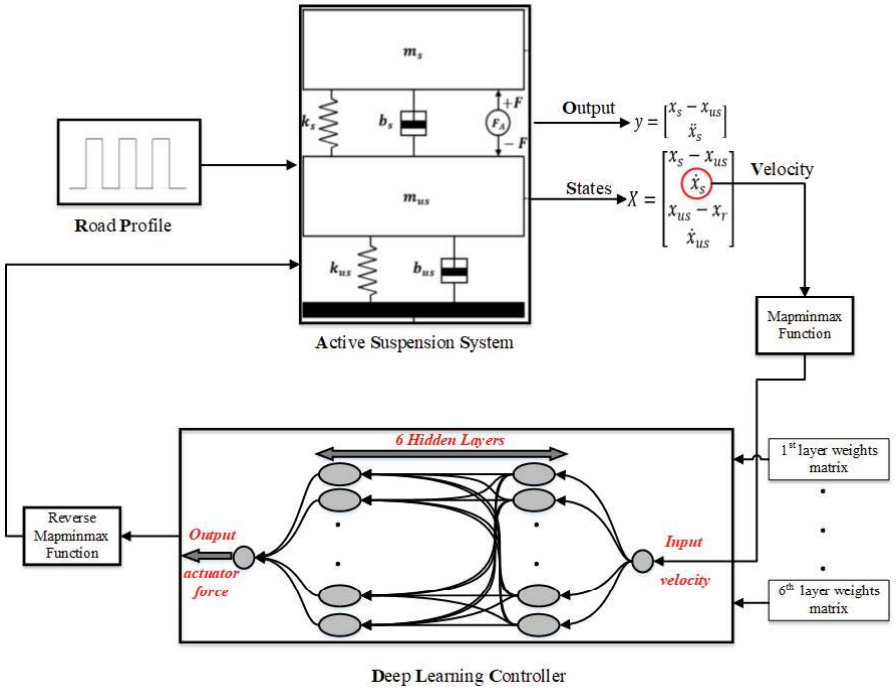

Figure 7. Model of the Proposed Deep Learning Controller Scheme

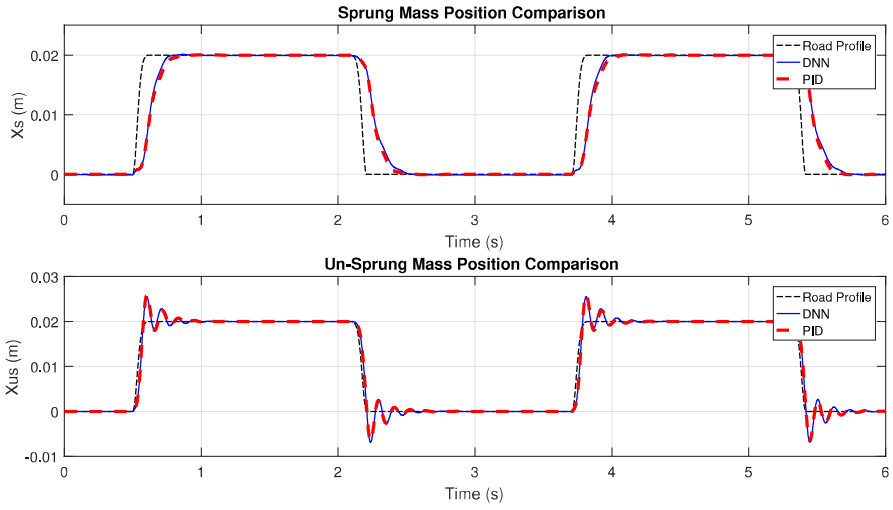

Figure 8. Sprung mass position comparison

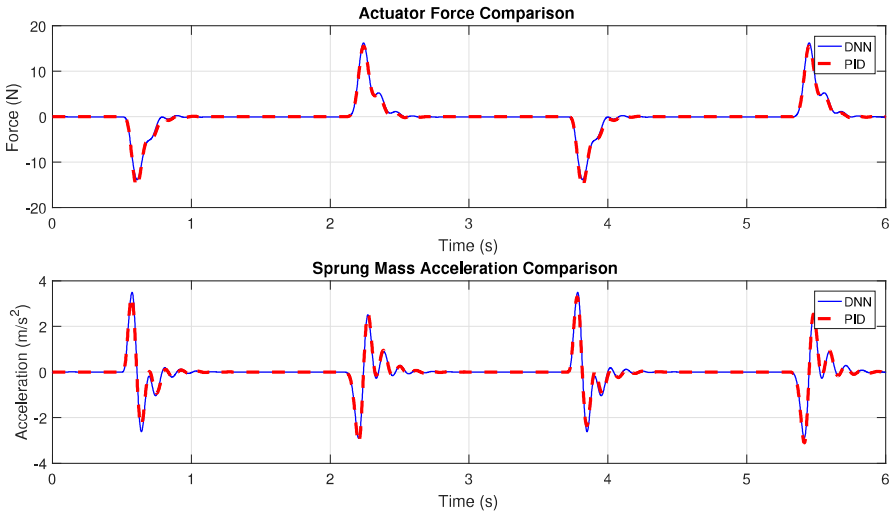

Figure 9. Actuator force comparison

a system such as this is suitable for electric vehicles. A typical vehicle can be operational for hours at a time and if the total energy consumption from an active suspension system is too high, it may be deemed useless in fully electric vehicles as it would negatively impacts its single-charge mileage range.

The sprung mass acceleration is the main parameter which affect passenger's comfort. Rapid changes in acceleration are more noticeable than the changes to velocity or displacement. Therefore, it is fundamental to minimise the overall vehicle body acceleration. The DNN Controller outperforms optimal PID controller for the peaks and the troughs as shown in Figure 12, providing a lower overall body acceleration. The average values obtained are $0.1032m/s^2$ for the DNN Controller and $0.1081m/s^2$ for the optimal PID controller. The DNN Controller shows a reduction of 4.5% in acceleration. Due to non-linearity of DNN, it deals with non-Gaussian noise better, hence outperforms the LQR controller when it is implemented in the Hardware in the loop system and tested with road profile recorded from the real car. This is consistent with Metered et al. (2010) which came to a similar conclusion although their suspension system was only semi-active that used a magnetic damper.

Discussion

The application of DL for training of DNNs requires implementation of different options through trial and error method. The initial choice of options is usually based on experience on similar structures and type of data.

Often DNN models run into the problem of *vanishing gradient*, when weights updates dW through backpropagation becomes ineffective due to gradients δ decreasing to very small values as they propagate or they can saturate due to the restrictive range of $Sigmoid$

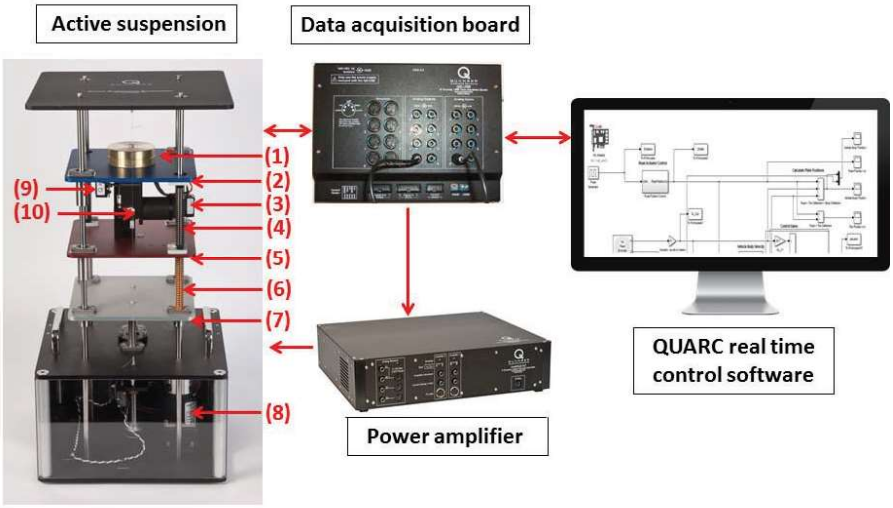

Figure 10. Diagram of the active suspension test rig:(1) Payload simulates passengers, (2) Blue plate simulates vehicle body mass, (3) Brushless DC motor produces control force, (4) Adjustable spring simulates vehicle suspension spring, (5) Red plate simulates vehicle wheel mass, (6) Adjustable spring simulates vehicle wheel stiffness, (7) White plate simulates the road, (8) Brushed DC motor simulates different road profiles, (9) Accelerometer, (10) High resolution encoder.

and $Tanh$ activation functions, being unable to reach all the layers. This means that Neural Network is not able to learn from the training data. The use of Rectified Linear Unit ($ReLU$) activation function has decreased the effect of vanishing gradient because it saturates only negative values and allows higher value of gradient to propagate, in fact the optimal structure contains several hidden layers with $ReLU$ in between layers with $Sigmoid$ or $Tanh$ activation functions.

Underfitting is a common issue that affected training where the DNN is unable to generalise the model and causes poor performance with both training and test data sets. Overcoming underfitting required tuning of training algorithm hyperparameters, such as learning rate, until suitable selection was found through a coarse to fine validation strategy, which starts from a wide range of values to understand their effect, and progressively reduces the range.

The use of trained DNNs as controller provides high adaptability level, given adequate data and training, it can easily be re-purposed to scale across different systems without significant changes in its structure and it allows to offload significant time and efforts spent in engineering and fine tuning of conventional controllers.

Linear state space model was implemented for PID with LQR although nonlinearity exists due to friction between components and wear of suspension system parts. DNN

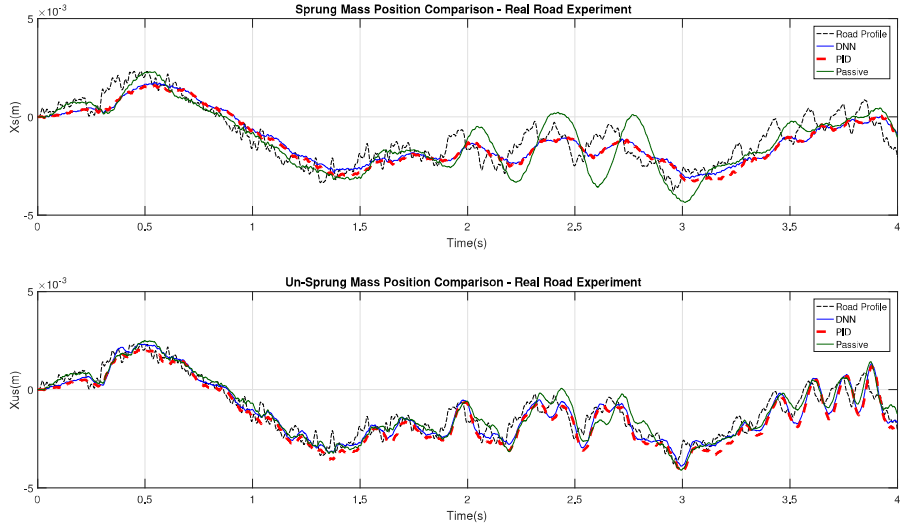

Figure 11. Rig Sprung and un-sprung mass positions with real road data

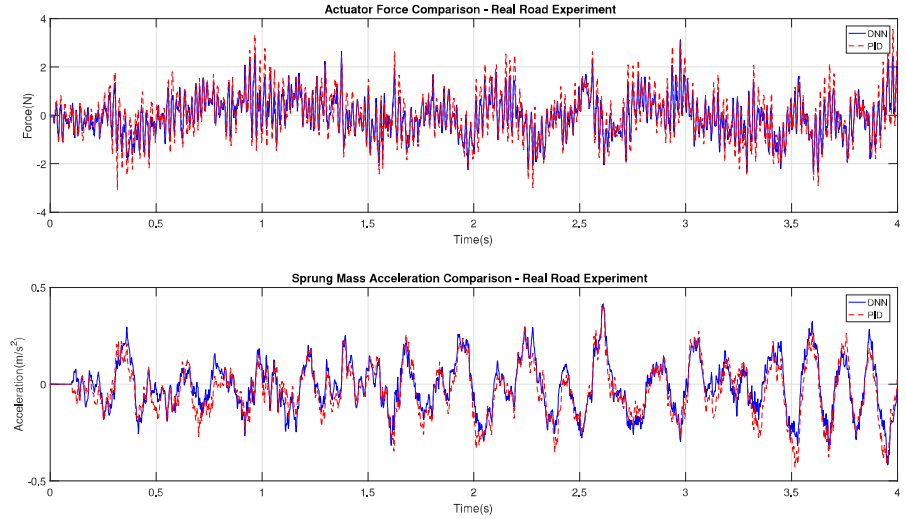

Figure 12. Rig actuator force and sprung mass acceleration with real road data

network outperforms PID controller with LQR due to inherited non-linearities, which allowed DNN to capture relevant dynamics trends of the system.

The majority of computational cost is paid during training offline so trained DNNs require low computational power, their processing time and memory remains the same regardless of the input. Therefore, they can be deployed in low cost mobile chips and

process complex data in almost real-time which cuts cost of implementation in real systems. The ease of implementation on the Quarc Active Suspension System Rig, which was not designed for DNNs, is a clear example.

Conclusion

In this paper, optimal structure of DNN controller for ASS was determined to have 5 hidden layers with 18 nodes per hidden layer while the activation functions are $Sigmoid$ for the input layer, $ReLU$, $Tanh$, $ReLU$, $Sigmoid$ and $ReLU$ for the output layers respectively. The developed DNN model has the ability to be implemented in real time and to be deployed in the HIL test.

The training of the optimal DNN structure was successful, the DNN Controller performed well in both simulation and experiments with real road profiles which have never been seen during the training process. The DNN controller has lower energy consumption than optimal PID controller while reducing the vehicle body acceleration which allows the DNN to outperform the optimal VP Controller under real conditions.

The performance improvement in terms of actuator force that was provided by the DNN controller can be explained by the fact that the it was trained with simulation data which did not present noise therefore, when deployed, the DNN gained an inherent filter which dynamically adjusts itself to provide a smoother and accurate control signal to the actuator. A 4% reduction in vehicle body acceleration might not be considered significant in comfort for the passengers but an average of 14% reduction in required actuator force, therefore in energy consumption, is a significant improvement considering the heavy and long term usage of this component, specially in the context of electric vehicles, in which energy saving is fundamental to increase autonomy. This results demonstrate a key point for the development of ASS, which is the fact that better performance can be achieved with lower energy consumption and force with the use of new techniques provided by DL.

Further improvement in the future include transition to suspension system from a quarter to a full car model, which will improve performance and handling, for example, effects of steering angle should be taken into account to compensate dynamic factors related to weight redistribution during cornering which will require improvements in the training methodology. Then, the developed enhanced controller should be implemented in car prototype. In addition to this, Long-Short-Term-Memory (LSTM) model could be a good candidate to be implemented in the future research. LSTM technique proved to be robust in modelling of long sequences data. This further development will lead to an important leap forward for autonomous cars, allowing AI to directly control and improve the suspension system performance.

Acknowledgement:

The authors would like to thank Mr. Dale Cooper for his help in acquiring the real road profile data from sensors placed on Renault Megan vehicle.

References

Alexandru C and Alexandru P (2011) A comparative analysis between the vehicles passive and active suspensions. *International Journal of Control, Automation and Systems*; 5(4): 371–378. URL https://pdfs.semanticscholar.org/2720/3f225577b141c4d27e698f7d93f7d6a87e1d.pdf.

Alkhoori F, Bin Safwan S, Zweiri Y, Sahinkaya MN and Seneviratne L 2017 PID-LQR Controllers for Quad-rotor Hovering Mode. In *2017 4rd International Conference on Systems and Informatics, ICSAI 2017*. ISBN 978-1-5386-1107-4, pp. 51–55.

Aly AA and Salem FA (2013) Vehicle suspension systems control: A review. *International Journal of Control, Automation and Systems*; 2(2): 46–54.

Bin Safwan S, Alkhoori F, Zweiri Y and Sahinkaya MN (2018) Implementation of a novel optimal PID methods in UAV applications. In *ICARM 2018 - IEEE International Conference on Advanced Robotics and Mechatronics, National University of Singapore, Singapore, July 2018*. ISBN 978-1-5386-1107-4.

Chen HY and Huang SJ (2005) Functional approximation-based adaptive sliding control with fuzzy compensation for an active suspension system. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*; 219(11): 1271–1280. DOI:10.1243/095440705X35026. URL https://doi.org/10.1243/095440705X35026. https://doi.org/10.1243/095440705X35026.

Dessort R and Chucholowski C (2017) Explicit model predictive control of semi-active suspension systems using artificial neural networks (ANN). *8th International Munich Chassis Symposium 2017*; : 207–228.

Fischer D and Isermann R (2004) Mechatronic semi-active and active vehicle suspensions. *Control Engineering Practice*; 12(11): 1353–1367. DOI:10.1016/j.conengprac.2003.08.003.

Gupta S, Ginoya D, Shendge PD and Phadke SB (2016) An inertial delay observer-based sliding mode control for active suspension systems. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*; 230(3): 352–370. DOI:10.1177/0954407015586076. URL https://doi.org/10.1177/0954407015586076. https://doi.org/10.1177/0954407015586076.

Heidari M and Homaei H (2013) Design a pid controller for suspension system by back propagation neural network. *Journal of Engineering*; 2013: 1–9. DOI:10.1155/2013/421543. URL https://www.hindawi.com/journals/je/2013/421543/.

Huang SJ and Lin WC (2007) A neural network based sliding mode controller for active vehicle suspension. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*; 221(11): 1381–1397. DOI:10.1243/09544070JAUTO242. URL https://doi.org/10.1243/09544070JAUTO242. https://doi.org/10.1243/09544070JAUTO242.

Kim C and Ro PI (1998) A sliding mode controller for vehicle active suspension systems with non-linearities. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*; 212(2): 79–92. DOI:10.1243/0954407981525812. URL https://doi.org/10.1243/0954407981525812. https://doi.org/10.1243/0954407981525812.

Metered H, Bonello P and Oyadiji SO (2010) An investigation into the use of neural networks for the semi-active control of a magnetorheologically damped vehicle suspension. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*; 224(7): 829–848. DOI:10.1243/09544070JAUTO1481. URL https://doi.org/10.1243/09544070JAUTO1481. https://doi.org/10.1243/09544070JAUTO1481.

Qin Y, Xiang C, Wang Z and Dong M (2018) Road excitation classification for semi-active suspension system based on system response. *Journal of Vibration and Control*; 24(13): 2732–2748. DOI:10.1177/1077546317693432. URL https://doi.org/10.1177/1077546317693432. https://doi.org/10.1177/1077546317693432.

Ramsbottom M, Crolla DA and Plummer AR (1999) Robust adaptive control of an active vehicle suspension system. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering* 1999; 213(1): 1–17. DOI:10.1243/0954407991526621. URL https://doi.org/10.1243/0954407991526621. https://doi.org/10.1243/0954407991526621.

Rashid U, Jamil M, Gilani S and Niazi I (2016) LQR based training of adaptive neuro-fuzzy controller. *Journal of Intelligent & Fuzzy Systems*; 54: 311–322. DOI:10.1007/978-3-319-33747-0_31.

Shin D, Lee G, Yi K and Noh K (2016) Motorized vehicle active suspension damper control with dynamic friction and actuator delay compensation for a better ride quality. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*; 230(8): 1074–1089. DOI:10.1177/0954407015598670. URL https://doi.org/10.1177/0954407015598670. https://doi.org/10.1177/0954407015598670.

Tsao YJ and Chen R (2001) The design of an active suspension force controller using genetic algorithms with maximum stroke constraints. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*; 215(3): 317–327. DOI:10.1243/0954407011525638. URL https://doi.org/10.1243/0954407011525638. https://doi.org/10.1243/0954407011525638.

Watton J, Holford KM and Surawattanawan P (2004) The application of a programmable servo controller to state control of an electrohydraulic active suspension. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*; 218(12): 1367–1377. DOI:10.1243/0954407042707650. URL https://doi.org/10.1243/0954407042707650. https://doi.org/10.1243/0954407042707650.

Zhao F, Dong M, Qin Y, Gu L and Guan J (2015) Adaptive neural networks control for camera stabilization with active suspension system. *Advances in Mechanical Engineering*; 7(8): 1–11. DOI:10.1177/1687814015599926. URL https://doi.org/10.1177/1687814015599926. https://doi.org/10.1177/1687814015599926.

Zhao F, Ge SS, Tu F, Qin Y and Dong M (2016) Adaptive neural network control for active suspension system with actuator saturation. *IET Control Theory Applications*; 10(14): 1696–1705. DOI:10.1049/iet-cta.2015.1317.