Neural Network based ACC for Optimized Safety and Comfort

Merry Cherian
PG Student
Department of Electrical and Electronics Engineering
Karunya University,Coimbatore

S.Paul Sathiyan
Assistant Professor
Department of Electrical and Electronics Engineering
Karunya University,Coimbatore

ABSTRACT
In recent years many studies on intelligent vehicles have been devoted to solve problem such as accident prevention, traffic flow smoothing. Adaptive Cruise Control (ACC) is used to maintain a constant safe distance between the host vehicle and the leading vehicle to avoid rear end collisions.It is an automotive feature that allows the speed of the vehicle to adapt to the traffic environment. ACC operates in distance control mode and velocity control mode. The method by which the ACC vehicle's speed is controlled is via engine throttle control and limited brake operation. The inter vehicular distance between the vehicle is measured. Desired speed is obtained from the distance measured. Neural Network Controller is trained to produce the desired acceleration and braking. In this paper, ACC is implemented using three types of Neural Network such as Back Propagation Network (BPN),Radial Basis Network (RBN) and Generalized Regression Neural Network (GRNN). Among the three it is observed that during safety conditions BPN tracks the speed better and during comfort conditions RBN acts best.

Keywords
Adaptive Cruise Control, Back Propagation Network, Radial Basis Network, Generalized Regression Neural Network.

1. INTRODUCTION
The Adaptive Cruise Control (ACC) System is an extension of the conventional Cruise Control System which is also known as Intelligent Cruise Control or Autonomous Intelligent Cruise Control (AICC) System. It overcomes the disadvantage of Conventional Cruise Control. This AICC system not only keeps the speed constant at a desired value but it adapts the speed according to the vehicle moving in front [1]. Hence, the system can be used in dense traffic with repeated start and stop situations [2]. The ACC automatically adjust the speed of the host vehicle to match the speed of the leading vehicle, which subsequently adjusts the distance between the two vehicles. If the preceding vehicle increases its speed, the ACC system of the host vehicle automatically increases its speed as well. A radar system attached to the front of the vehicle is used to detect whether slower moving vehicles are in the ACC vehicle's path. If a slower moving vehicle is detected, the ACC system will slow the host vehicle down and control the clearance, or time gap between the host vehicle and the forward vehicle. If the system detects no vehicle in the front, then the ACC system will accelerate the vehicle to its set speed. This operation allows the ACC

vehicle to autonomously slow down and speed up with traffic without intervention from the driver. The method by which the ACC vehicle's speed is controlled is via engine throttle control and limited brake operation. The distance and speed of the vehicle is measured. Desired speed is obtained from the distance measured [3]. Based on the desired speed and actual speed, speed error is calculated. In this paper ACC is implemented in the vehicle model using three Neural Networks. They are Back Propagation Network (BPN), Radial Basis Network (RBN) , Generalized Regression Neural Network (GRNN) and performance analysis is done among the three.

2. BPN BASED ACC
Neural networks are composed of simple elements operating in parallel. These elements are inspired by biological nervous systems. As in nature, the network function is determined largely by the connections between elements. You can train a neural network to perform a particular function by adjusting the values of the connections (weights) between elements. Commonly neural networks are adjusted, or trained, so that a particular input leads to a specific target output.Back Propagation Network is a feed forward neural network. It was created by generalizing the Widrow-Hoff learning rule to multiple-layer networks and nonlinear differentiable transfer functions. Input vectors and the corresponding target vectors are used to train a network until it can approximate a function, associate input vectors with specific output vectors, or classify input vectors in an appropriate way as defined. Networks with biases, a sigmoid layer, and a linear output layer are capable of approximating any function with a finite number of discontinuities. Standard back propagation is a gradient descent algorithm, as is the Widrow-Hoff learning rule, in which the network weights are moved along the negative of the gradient of the performance function. The term back propagation refers to the manner in which the gradient is computed for nonlinear multilayer networks. There are a number of variations on the basic algorithm that are based on other standard optimization techniques, such as conjugate gradient and Newton methods. Neural network toolbox implements a number of these variations.

2.1 Architecture of BPN

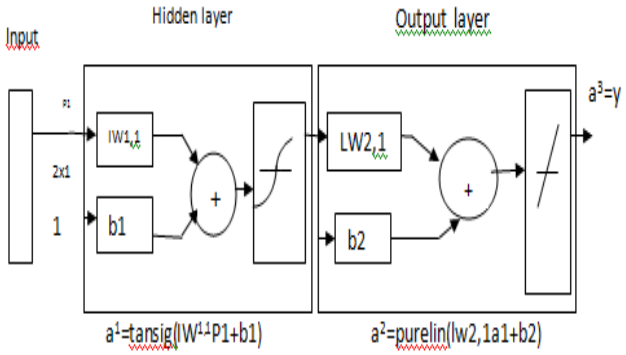

Fig 1: Architecture of BPN

Feedforward networks often have one or more hidden layers of sigmoid neurons followed by an output layer of linear neurons. Multiple layers of neurons with nonlinear transfer functions allow the network to learn nonlinear and linear relationships between input and output vectors. The linear output layer lets the network produce values outside the range –1 to +1. On the other hand, if you want to constrain the outputs of a network (such as between 0 and 1), then the output layer should use a sigmoid transfer function (such as logsig). Multilayer networks often use the log-sigmoid transfer function logsig or tan-sigmoid function tansig.

2.2 Algorithm of BPN

STEP 1: Creating a training set of input speed error and targets throttle command and brake command [4].

STEP 2: Creating a feedforward network, with 2 layers.

Hidden layer-3 Neurons, Tansig transfer function

Output layer-1 Neuron, Purelin transfer function

STEP 3: Training function: Levenberg-Marquardt

STEP 4: Setting of training parameters

Learning rate=0.05

Error goal=0

STEP 5: Training the network.

STEP 6: Finally, outputs are simulated.

3. RBN BASED ACC

Radial basis networks can require more neurons than standard feedforward backpropagation networks, but often they can be designed in a fraction of the time it takes to train standard feedforward networks. They work best when many training vectors are available. A Radial Basis Function (RBF) neural network has an input layer, a hidden layer and an output layer. The neurons in the hidden layer contain Gaussian transfer functions whose outputs are inversely proportional to the distance from the center of the neuron.

3.1 Architecture of RBN

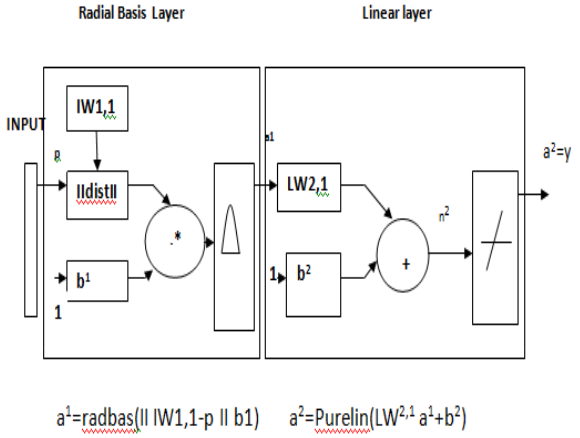

Fig 2: Architecture of RBN

The || dist || box in this figure accepts the input vector **p** and the input weight matrix **IW**1,1, and produces a vector having *S*1 elements. The elements are the distances between the input vector and vectors **IW**1,1 formed from the rows of the input weight matrix. The bias vector **b**1 and the output of || dist || are combined with the MATLAB operation .*****, which does element-by-element multiplication. The function newrb iteratively creates a radial basis network one neuron at a time. Neurons are added to the network until the sum-squared error falls beneath an error goal or a maximum number of neurons have been reached. The call for this function is

net = newrb(P,T,GOAL,SPREAD)

The function newrb takes matrices of input and target vectors P and T, and design parameters GOAL and SPREAD, and returns the desired network. The design method of newrb is similar to that of newrbe. The difference is that newrb creates neurons one at a time. At each iteration, the input vector those results in lowering the network error the most is used to create a radbas neuron. The error of the new network is checked, and if low enough newrb is finished. Otherwise the next neuron is added. This procedure is repeated until the error goal is met or the maximum number of neurons is reached. We can design radial basis networks with the function newrbe. This function can produce a network with zero error on training vectors. It is called in the following way:

net = newrbe(P,T,SPREAD)

The function newrbe takes matrices of input vectors P and target vectors T, and a spread constant SPREAD for the radial basis layer, and returns a network with weights and biases such that the outputs are exactly T when the inputs are P.

3.2 Algorithm of RBN

STEP 1: Creating a training set of input speed error and targets throttle command [5] and brake command**.**

STEP 2: Plotting of training vectors.

STEP 3: Finding a function which fits the data points [6], done by Radial Basis network.

- Number of layers: 2
- Hidden layer-Radial Basis neurons.

- Output layer-Linear neurons.

STEP 4: Radial Basis transfer function is defined and plotted.

STEP 5: Three Radial Basis functions are scaled and summed to produce a function.

STEP 6: Creating a Radial Basis network with function newrb.

- eg=0.02
- Spread=0.1

STEP 7: Finally, simulating the network response.

4. GRNN BASED ACC

Generalized Regression networks are variant of Radial Basis network. A GRNN is often used for function approximation. It has a radial basis layer and a special linear layer. For problems with small to medium size training sets, networks are usually more accurate than RBF network's.

4.1 Architecture of GRNN

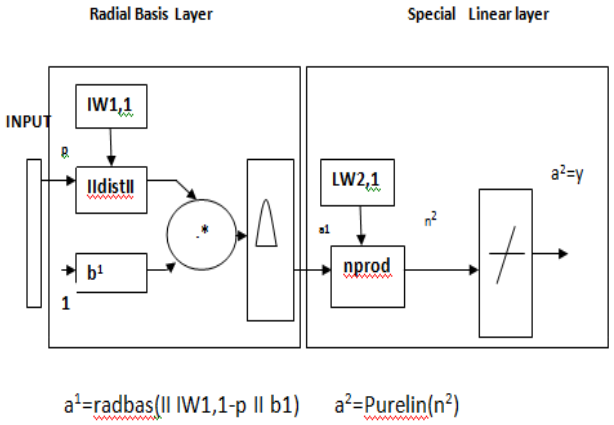

Fig. 3 Architecture of GRNN

4.2 Algorithm of GRNN

STEP 1:Creating a training set of input speed error and targets throttle command and brake command

STEP 2:Creating a GRNN network by newgrnn with arguments input P, targets T, spread constant.

- Number of layers:2
- Hidden layer: Radial Basis layer
- Output layer: special linear layer

STEP 3:Finally simulating the network response.

5. SIMULATION RESULTS

The Fig. 4 describes the vehicle model used for simulation purpose. In this model as throttle changes, gear and clutch sequence changes and corresponding speed is obtained. The throttle command is given by means of neural network controller according to speed error. Three neural network controller such as BPN,RBN,GRNN are used.

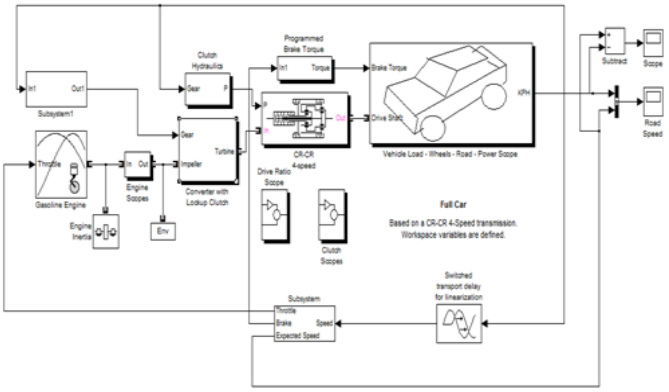

Fig. 4 Vehicle model

The architecture for the GRNN is shown above. It is similar to the radial basis network, but has a slightly different second layer. Here the nprod box shown above (code function normprod) produces *S*2 elements in vector n2. Each element is the dot product of a row of LW2,1 and the input vector a1, all normalized by the sum of the elements of a1. The first layer has many neurons as there are input/ target vectors in P. Specifically, the first-layer weights are set to P'. The bias b1 is set to a column vector of 0.8326/SPREAD. The user chooses SPREAD, the distance an input vector must be from a neuron's weight vector to be 0.5.Each neuron's weighted input is the distance between the input vector and its weight vector, calculated with dist. Each neuron's net input is the product of its weighted input with its bias, calculated with netprod. Each neuron's output is its net input passed through radbas. If a neuron's weight vector is equal to the input vector (transposed), its weighted input will be 0, its net input will be 0, and its output will be 1. If a neuron's weight vector is a distance of spread from the input vector, its weighted input will be spread, and its net input will be sqrt(-log(.5)) (or 0.8326). Therefore its output will be 0.5. Function newgrnn can be used to create a GRNN. Now obtain a GRNN with

net = newgrnn(P,T)

Fig. 5 describes the subsystem of the ACC using neural network controller**.**The distance between host and lead vehicle is given to the lookup table where the expected speed of the host vehicle corresponding to the distance is being given.Based on this,the expected speed(speed of lead vehicle) corresponding to the distance is obtained.This speed is then compared with the actual speed of the host vehicle and speed error is obtained. Thus input to the Neural Network controller is speed error.

$$S_{error} = S_{lead} - S_{actual} \quad (1)$$

The neural network controller is trained in such a way that it will produce the desired acceleration and braking corresponding to the speed error [7]-[12]. The output of the Neural Network controller is fed to the vehicle and corresponding speed is obtained.

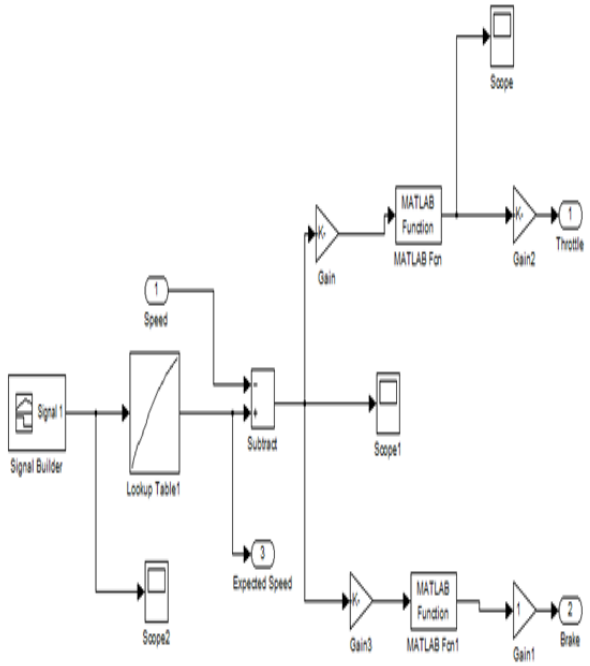

Fig. 5 Subsystem of ACC using neural network controller.

5.1 Calculation of throttle

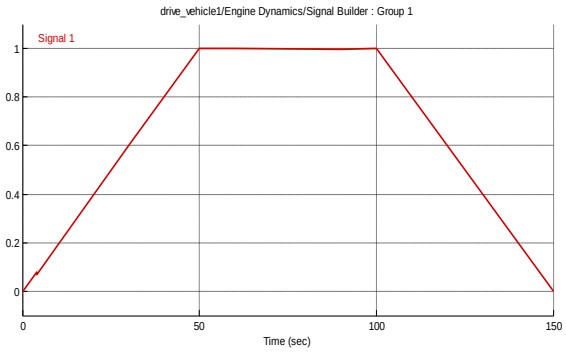

Fig. 6 Graph of variable throttle.

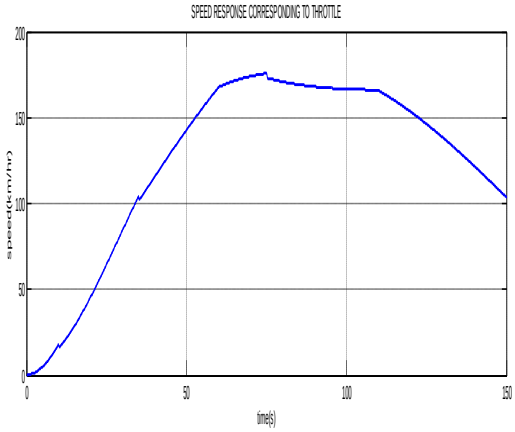

Fig. 7 Speed response corresponding to variable throttle.

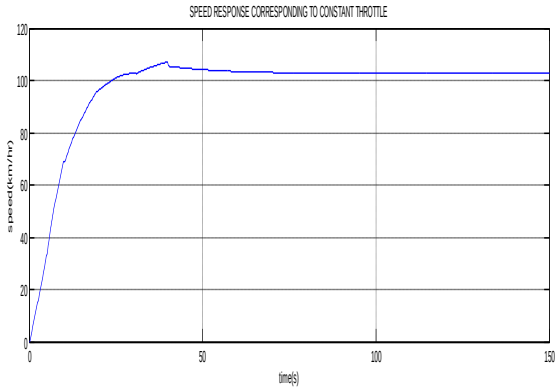

Fig. 8 Speed response corresponding to steady throttle

Fig. 6 describes the graph of variable throttle.Based on the variable throttle, corresponding variable speed is shown in Fig 7.Fig. 8 describes the constant speed response corresponding to constant throttle. Based on the above study corresponding throttle valve is obtained for different speeds.This throttle value is used for training neural network controller.

5.2 Performance comparison of ACC using three neural network controllers

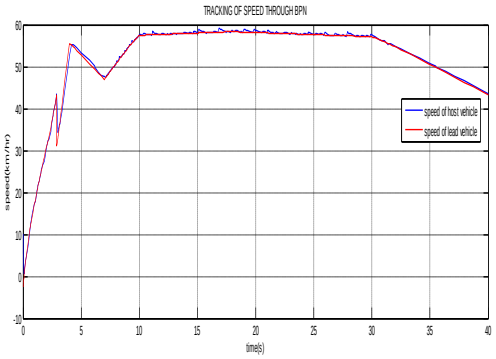

Fig. 9 Tracking of speed through BPN

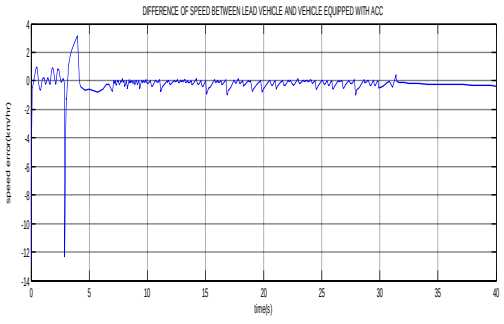

Fig. 10 Speed error between lead and host vehicle while using BPN

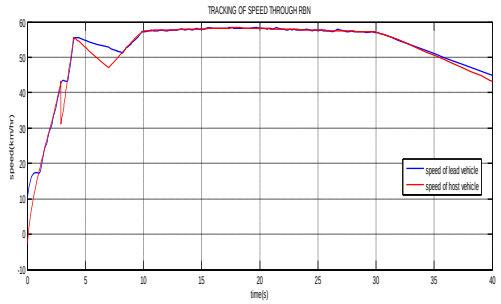

Fig. 11 Tracking of speed through RBN

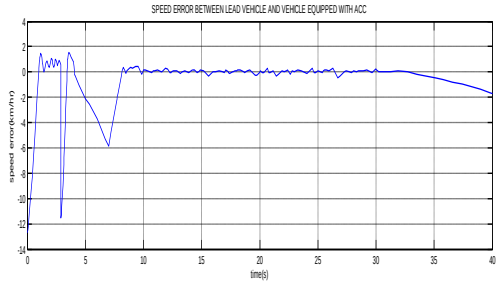

Fig. 12 Speed error between lead and host vehicle while using RBN

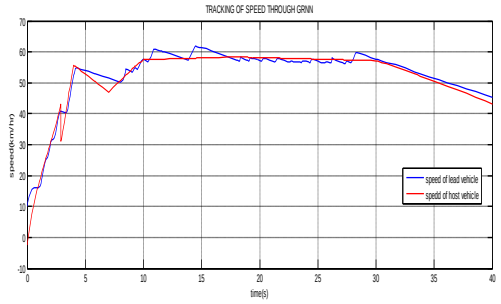

Fig. 13 Tracking of speed through GRNN

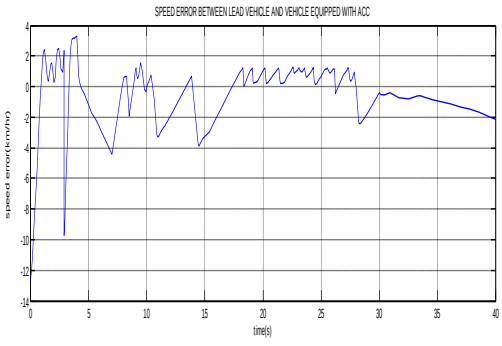

Fig. 14 Speed error between lead and host vehicle while using GRNN

Fig. 9,11 and 13 shows the tracking of speed through three different controllers such as BPN, RBN and GRNN. It is observed that during sudden acceleration and deceleration of the lead vehicle speed, BPN is producing the best result when compared to the other networks. i.e., during the safety conditions of the vehicle, BPN is tracking the speed with minimum error which is observed in Fig 10. When speed is tracked through RBN, it is observed from Fig 12. that during comfort conditions i.e., when the lead vehicle speed is slowly increasing and decreasing or during the times of constant speed, RBN is acting as the best when compared to the other networks. i.e., during the comfort conditions of the vehicle RBN is tracking the speed with minimum error. Table I. shows the performance comparison of ACC using three different controllers.

Table I. Comparative Study Of Different Controllers

Name of controller	Speed deviation (km/hr)	Safety	Comfort
BPN	6.8	✔	
RBN	9.9		✔
GRNN	11.8		

6. CONCLUSION

The three mentioned neural network were used with the vehicle model for simulating the Adaptive Cruise Control (ACC). The simulation was performed in Matlab. Among the three network , during safety conditions (sudden increase and decrease of speed) BPN acts as the best and during comfort conditions (smooth driving) RBN acts as the best. In future it is planned to implement the same in real time for validating the simulation results.

7. REFERENCES

[1] Sakda Panwai, and Hussein Dia, "Neural Agent Car-following Models,"*IEEE Trans. Intell. Transp. Syst.*, vol 8,no. 1,pp. 60-70,March 2007.

[2] P.Venhovens, K.Naab and B.Adiprasito "Stop and Go Cruise Control",*International Journal of Automotive Technology*, Vol.1, no.2, pp.61-69.2000

[3] John-Jairo Martinez and Carlos Canudas-de-Wit "A Safe Longitudinal Control for Adaptive Cruise Control and Stop-and-Go Scenarios" *IEEE Trans.Control Systems Technology*, vol. 15, no.2, pp.246-258 March.2007

[4] Sungwoo CHOI, Brigitte d'andr' ca-novel, Michel FLIESS, Hugues Mounier, Jorge VILLAGRA, "Model-free control of automotive engine and brake for Stop-and-Go scenarios," *European control conference*, 2009

[5] Luke Ng, Christopher M. Clark, and Jan P. Huissoon, "Reinforcement Learning of Adaptive Longitudinal Vehicle Control for Dynamic Collaborative Driving*" proceedings of the IEEE Intelligent Vehicles Symposium*, Eindhoven University of Technology Eindhoven, The Netherlands, June 4-6,2008

[6] Yang Bin, Keqieng Li, Xiaomin Lian Hiroshi Ukawa, Masatoshi Handa, Hideyuki Idonuma "Longitudinal Acceleration Tracking Control of Vehicular Stop-and-go Cruise Control System", *Proceedings of the 2004 IEEE Int. Conf .Networking, Sensing & Control* pp 607-612, Taiwan, March 21-23,2004

[7] Kyongsu Yi, Ilki Moon and Young Do Kwon "A Vehicle-to-Vehicle Distance Control Algorithm for stop Stop-and-go Control", in *Conf. Rec.2001 IEEE. Conf. Intelligent Transportation Systems*, pp 478-482

[8] Jose' E. Naranjo, Carlos Gonzalez, Member, IEEE, Ricardo Garcia, and Teresa de Pedro "ACC+Stop&Go Maneuvers with the Throttle and Brake Fuzzy Control",*IEEE Trans.Intelligent Transportation Systems.*, vol.7, no. 2,pp 213-225,June 2006

[9] Nassaree Benalie, Worrawut Pananuruk, Somphong Thanok, and Manukid Parnichkun "Improvement of Adaptive Cruise Control System based on Speed Characteristics and Time Headway*" IEEE/RSJ Int. Conf on Intelligent Robots and Systems.*, pp.2403-2408 October 2009.

[10] S. Paul Sathiyan and A. Wisemin lins,"Soft Computing Based Adaptive Cruise Control",*Indian Journal Of Computer Science and Engineering(IJSCE)*,vol 2,no. 1,pp. 68-76,2011

[11] Vicente Milanes, Jorge Villagra, Jorge Godoy, and Carlos Gonzalez, Member, IEEE," Comparing Fuzzy and Intelligent PI Controllers in Stop-and-Go Manoeuvres",*IEEE Trans. On Control Systems Technology*,pp. 1-9,2011.

[12] Rudwan Abdullah,Amir Hussain,Kevin Warwick,Ali Zayed, "Autonomous intelligent cruise control using a novel multiple-controller framework incorporating fuzzy-logic-based switching and tuning",ELSEVIER(2008),pp. 2727-2741.