International Journal of Technology 11(7) 1451-1462 (2020)
Received July 2020 / Revised December 2020 / Accepted December 2020

International Journal of Technology

http://ijtech.eng.ui.ac.id

Neural Network Predictive Control Approach Design for Adaptive Cruise Control

Pratama Mahadika[1], Aries Subiantoro[1*], Benyamin Kusumoputro[1]

[1]*Department of Electrical Engineering, Faculty of Engineering, Universitas Indonesia, Kampus UI Depok, Depok 16424, Indonesia*

Abstract. As one part of the advanced driver assistance systems (ADAS), adaptive cruise control (ACC) is introduced to reduce the possibility of traffic accidents by controlling the throttle and the pressure on the brakes to maintain a safe distance from the vehicle in front. Generally, linearized model-based controllers are used in the ACC. In this paper, a new approach to ACC's inner loop is developed by designing the controller using neural network predictive control (NNPC) which integrates the capability of artificial neural networks (ANN) to imitate vehicle characteristics and model predictive control (MPC) to obtain the minimized quadratic error between future reference trajectories and predicted outputs. Two separate control loops will be used: an outer loop based on a decision algorithm, and the PI controller, which will give the inner loop a speed reference to maintain the safe distance from the vehicle in front. NNPC is used in the inner loop to manipulate throttle and brake pressure on the brakes in order to control the speed of the following vehicle. Simulations will be carried out using software-in-the-loop (SIL) between CarSim and Simulink. The ANN model is identified and verified to mimic the nonlinearity behavior of the vehicle model using the mean square error (MSE) parameter. The results of this study are that the ANN model is able to imitate the vehicle dynamic with MSE equal to 0.0095, and the controller can maintain a safe distance while having a smooth response.

Keywords: Adaptive cruise control; Artificial neural network; Dynamic vehicle model; Neural network predictive control

1. Introduction

In recent years, trends in improving driving safety have become an important concern for the automotive industry because traffic accidents are major concerns faced by drivers. These problems can be avoided by introducing some forms of driver assistance to prevent accidents. In fact, 50% of accidents that occur are rear-end collisions. That's why advanced driver assistance systems (ADAS) are developed by automobile manufacturers to make driving safer. The National Transportation Safety Board said that active safety systems are 50% more effective in reducing death rates in accidents compared to passive safety systems such as airbags (ACEA, 2018). As part of the ADAS system, active cruise control (ACC) was developed in early 1990. The ACC system is capable of adjusting vehicle speed while maintaining a safe distance from the vehicle in front. This system modulates the throttle valve and brake pressure to reduce or accelerate the vehicle to the desired speed and

*Corresponding author's email: aries.subiantoro@ui.ac.id, Tel.: +62-21-7270078; Fax: +62-21-7270077
doi: 10.14716/ijtech.v11i7.4592

distance. Radar, laser, and other sensory devices are used to measure the distance from vehicles in front, so the ACC system can choose a proper driving mode.

Several methods for the ACC system have been developed in four-wheeled vehicles. Classical methods such as PID and fuzzy have been developed since a few decades ago. Rout and his colleagues utilized a PID controller that was optimized with genetic algorithm (GA) to produce optimal PID tuning (Rout et al., 2016). Shakouri developed the ACC system with the gain-scheduling method using PI and LQ controllers to manipulate throttle valve and brake pressure (Shakouri et al., 2011), and Pananurak and his colleagues used fuzzy logic algorithms for ACC systems (Pananurak et al., 2009). Some predictive methods began to be developed because they resulted in better control of vehicle dynamics. Shakouri also developed the two-loop ACC system by utilizing the model predictive control (MPC) for throttle and brake control as inner loop control and PI as a speed controller for outer-loop control (Shakouri and Ordys, 2014). After that, Naus and his colleagues utilized implicit MPC and multi-parametric quadratic programs for online identification of ACC systems (Naus et al., 2010). Miftakhudin and colleagues developed a multistage MPC system with constraints for the ACC controller to achieve a smooth response (Miftakhudin et al., 2019). On problem shared by all the research mentioned above, is that the controller uses the linearization method in modeling the longitudinal motion of four-wheeled vehicles. These methods limit the controller's ability to work only in a specified range and are difficult to obtain for a large working range.

From many control methods that have been developed, artificial neural network (ANN) has not been widely applied in automotive controllers, especially in ACC systems, even though ANN is widely known for its ability to capture nonlinear phenomena. For that reason, the main contribution of this work is to make a controller that integrates the ability of ANN to capture nonlinear dynamics of moving vehicles and the predictive ability of MPC to control ACC systems. This method, called neural network predictive control (NNPC), began development 1996 when Soloway started working on neural generalized predictive control that combines ANN for model identification and generalized predictive control (GPC) for the controller (Soloway and Haley, 1996). These methods created a new problem for minimization of the cost function in GPC. Newton-Raphson was used to compute optimization problems numerically to obtain optimal control sequences for the controller. This paper used a slightly modified method, quasi-Newton, to compute the control sequence for the controller and Broyden Fletcher Goldfarb Shanno (BFGS) algorithm to solve inverse Hessian matrices that appear in Newton-based optimization. In this research, simulation will be carried out in software-in-the-loop (SIL) between MATLAB and CarSim.

2. Methods

In this research, a combination of ANN and MPC will be used to control the speed and distance of a vehicle. A dynamic vehicle is used, and CarSim will provide its model and behavior similar to the real-life movement of a vehicle. NNPC will then be induced to the system as ACC. The dynamics of a vehicle model and NNPC are described in the following section.

2.1. Dynamic Vehicle Model

The model used in this research has one degree of freedom by taking the center of gravity in the center of the vehicle. There will be two vehicles: the one that follows (the follower) as the vehicle being controlled and the one in front (the leader) as disturbance and the speed reference of the vehicle that follows. The dynamic model of the vehicle used is based on Newton's second law by considering various external factors from the environment as follows:

$$m\dot{v} = \frac{1}{r}\left[R_{tr}R_f C_{tr}\left(\frac{N_e}{K_{tc}}\right)^2 + T_b\right] - \frac{1}{2}\rho A C_d v^2 - C_r mgcos(\theta) + mgsin(\theta) \quad (1)$$

where m denotes vehicle's speed, $\dot{v}$ denotes acceleration, r is wheel radius, R_{tr} and R_f indicate gear ratio and the final gear ratio respectively, C_{tr} is torque ratio, N_e denotes engine speed, k_{tc} is a capacity factor, T_b is brake torque, ρ is the air density, A is the front cross-sectional area of the vehicle, C_d is the drag coefficient, C_r is the rolling resistance coefficient, g is gravity, and θ indicates the slope of the road.

The first equation of the function above is the torque function sent to the wheels (T_w), the second equation shows the aerodynamic function of the vehicle, the third function shows the vehicle's rolling resistance force, and the fourth equation shows the function due to gravity. In this study, the gravitational force in the fourth equation is not taken into account because the scenario used does not experience elevation changes. In the first equation, a further equation can be explained as follows:

$$T_w = R_f R_{tr} C_{tr}(T_e(u_t, N_e) - I_{ei}\dot{N}_e \quad (2)$$

where T_e is the torque produced by the engine, u_t indicates throttle valve percentage and I_{ei} is the summation of the engine and impeller moment of inertia. From this equation, there is another nonlinear function in terms of torque produced by the engine where the function is in the form of an engine map. This engine map is hard to formulate in a mathematical equation. Therefore, ANN is used to identify vehicle longitudinal motion. The desired safe distance used in this research is based on constant-time headway policy as follows (van den Bleek, 2007):

$$d_{des} = l + d_s + T_h v_{host} \quad (3)$$

where d_{des} is the desired safe distance, l is the length of the vehicle, d_s is the additional distance between vehicles, v_{host} is the user's vehicle speed, and T_h is the constant-time headway obtained from the estimate of the human reaction time (Martinez and Canudas-de-Wit, 2007).

2.2. *NN-Based Nonlinear Identification*

To identify vehicle longitudinal motion using ANN, a neural network auto regressive with exogenous (NNARX) model can be used. The one-step-ahead predictions are given as follows:

$$\hat{y}(t) = g\big(y(t-1), \dots, y(t-n), u(t-1), \dots, u(t-m)\big) \quad (4)$$

where g is a non-linear function of the system modeled by a neural network, n is the number of past outputs used, m is the number of past inputs used, and k is the time of each step. For the learning process, we need input and output data for ANN. Random vehicle control signals will be sent to the vehicle model in CarSim as input data, and vehicle speed due to the control signals will be saved as output data. All the data gathered from CarSim will be used to train the ANN model in a MATLAB environment. In the training process, Figure 1a can be used to train the ANN model.

An ANN model with a three-time delay for each input-output, seven hidden-layer units, and a single output will be used in this research. The structure of ANN used to identify the vehicle model can be seen in Figure 1b.

2.3. Controller Structure

In this paper, the ACC based on NNPC was developed to regulate the distance and speed between host and lead vehicles by calculating the optimal predicted control signal through the receding horizon. The control structure consists of an inner loop and an outer loop, as shown in Figure 2, with $v_{rel} = v_{fol} - v_{lead}$. The outer loop uses a PI controller and was developed in order to calculate the desired velocity. This set point will be tracked by NNPC by manipulating the engine torque and the brake torque.

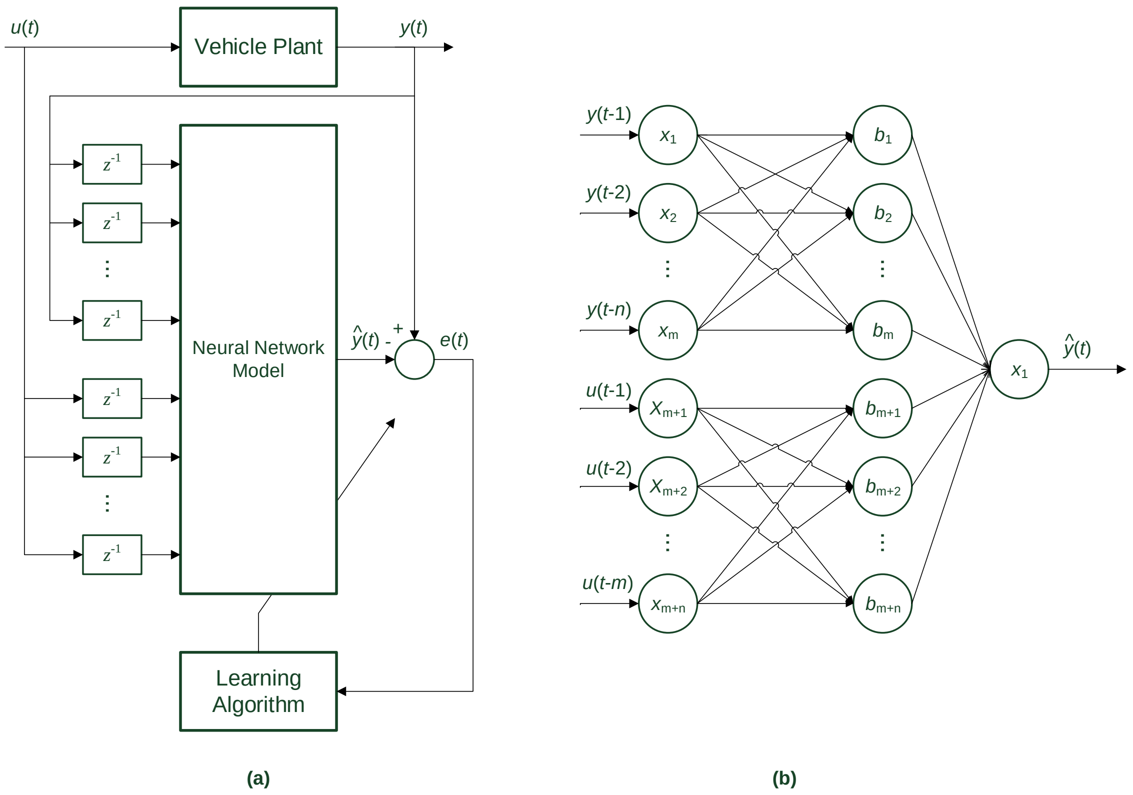


Figure 1 Training process: (a) Block diagram for the training process; (b) ANN structure used in the training process

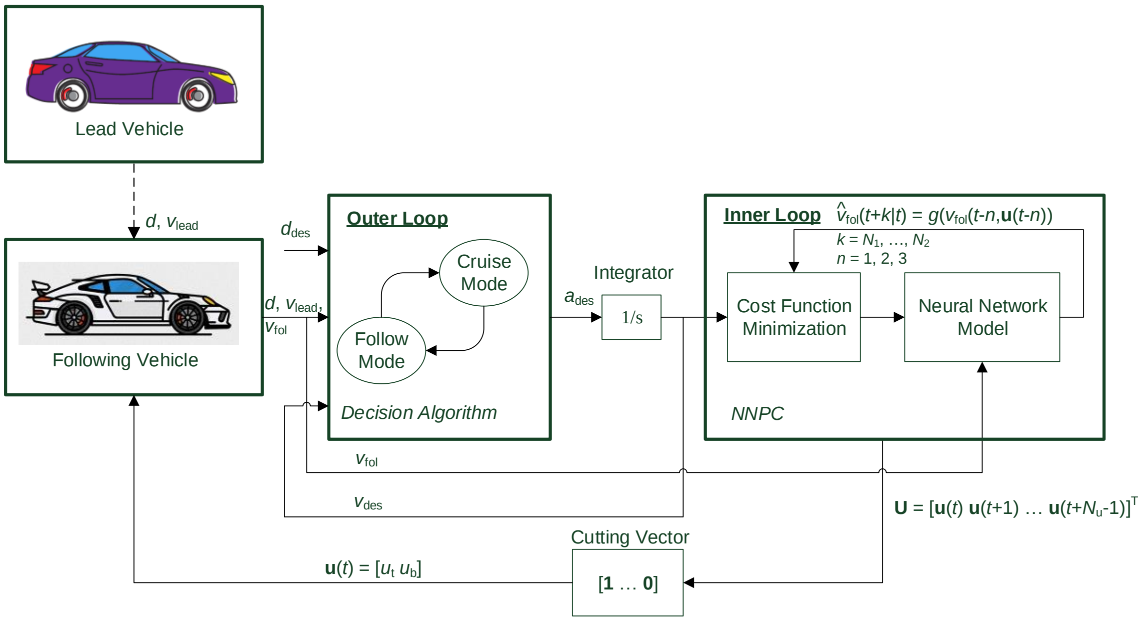


Figure 2 Block diagram for ACC system with NNPC

NNPC is basically an MPC that uses an ANN model to compute plant output predictions; hence, it has the same cost function as conventional MPC, but has different approaches to find output predictions and cost function minimization. The cost function used in NNPC can be calculated by the following equation:

$$J = \sum_{i=N_1}^{N_2} [r(t+i) - \hat{y}(t+i)]^2 + \rho \sum_{i=1}^{N_u} \Delta u(t+i-1)^2 \tag{5}$$

where J indicates the value of the minimization function, N_1 is the minimum cost horizon, N_2 is the prediction or maximum cost horizon, N_u is the control horizon, $r(t)$ is the reference signal, $\hat{y}(t)$ is the predicted result of the output, ρ is the weighting factor, which provides a penalty function for changing the control signal.

The output of the NN model is given by the following equation:

$$\hat{y}(t) = \sum_{j=1}^{n_h} w_j f_j\left(\theta_j(t)\right) + b \tag{6}$$

where n_h is the number of nodes in the hidden layer, and:

$$\theta(t) = \sum_{j=1}^{m} w_{j,i} u\,(t-i) + \sum_{j=1}^{n} w_{j,m+1+i} y\,(t-i) + b_j \tag{7}$$

The NNPC predicts the future output for a determined prediction horizon N_2 at each instant *t,* based on the known values of past inputs and outputs and on the future control signals, described as:

$$\hat{y}(t+k) = \sum_{j=1}^{n_h} w_j f_j\left(\theta_j(t+k)\right) + b \tag{8}$$

where:

$$\theta(t) = \begin{cases} \sum_{j=1}^{m} w_{j,i} u\,(t-i) u(t+k-i) + \vartheta_j + \varphi_j + b_j & , k - N_u < i \\ \sum_{j=1}^{m} w_{j,i} u\,(t-i) u(t+N_u) + \vartheta_j + \varphi_j + b_j & , k - N_u \geq i \end{cases} \tag{9}$$

and:

$$\begin{aligned} \vartheta_j &= \sum_{i=1}^{\min(k,n)} w_{j,m+1+i} y\,(t+k-i) \\ \varphi_j &= \sum_{j=k+1}^{n} w_{j,m+1+i} y\,(t+k-i) \end{aligned} \tag{10}$$

The minimization of cost function J is performed at each time step to produce a sequence of future control signals $\mathbf{U} = [\mathbf{u}(t)\ \mathbf{u}(t+1)\ \ldots\ \mathbf{u}(t+N_u-1]^T$. From this sequence, only the first control signal $\mathbf{u}(t)$ is given to the system. In order to find the optimal control signal at each iteration, the quasi-Newton algorithm will be used, as in Sørensen et al. (1999). The iterative method is used to find the optimal control signal in the following equation:

$$\mathbf{U}^{(i+1)} = \mathbf{U}^{(i)} + \mu^{(i)} \mathbf{F}^{(i)} \tag{11}$$

where $\mathbf{U}^{(i)}$ indicates the current iteration of the sequence of future control inputs, $\mu^{(i)}$ is step size, and $\mathbf{F}^{(i)}$ is the search direction. The search direction used in Equation 11 is calculated by:

$$\mathbf{F}^{(i)} = -\mathbf{B}^{(i)}\mathbf{G}\left(\mathbf{U}^{(i)}(t)\right) \tag{12}$$

where $\mathbf{B}^{(i)}$ specifies the approximating inverse Hessian according to the BFGS algorithm (Chong and Zak, 2013):

$$\begin{aligned}\mathbf{B}^{(i+1)} \quad = \quad & \mathbf{B}^{(i)} + \left(1 + \frac{\Delta\mathbf{G}^{(i)T}\mathbf{B}^{(i)}\Delta\mathbf{G}^{(i)}}{\Delta\mathbf{G}^{(i)T}\Delta\mathbf{U}^{(i)}}\right)\frac{\Delta\mathbf{U}^{(i)}\Delta\mathbf{U}^{(i)T}}{\Delta\mathbf{U}^{(i)T}\Delta\mathbf{G}^{(i)}} \\ & - \frac{\mathbf{B}^{(i)}\Delta\mathbf{G}^{(i)}\Delta\mathbf{U}^{(i)T} + \left(\mathbf{B}^{(i)}\Delta\mathbf{G}^{(i)}\Delta\mathbf{U}^{(i)T}\right)^{T}}{\Delta\mathbf{G}^{(i)T}\mathbf{U}^{(i)}}\end{aligned} \tag{13}$$

with $\Delta\mathbf{G}(i) = \mathbf{G}(i+1) - \mathbf{G}(i)$, $\Delta\mathbf{U}(i) = \mathbf{U}(i+1) - \mathbf{U}(i)$, and $\mathbf{G}\left(\mathbf{U}^{(i)}(t)\right)$ indicates the gradient of the cost function or Jacobian with respect to the control inputs. By differentiating the predicted output in Equation 8 with respect to $\mathbf{u}(t+h)$, the hth elements of the Jacobian can be obtained as follows:

$$\frac{\partial y(t+k)}{\partial u(t+h)} = \sum_{j=1}^{n_h} w_j \frac{\partial f\left(\theta_j(t+k)\right)}{\partial u(t+h)} \tag{14}$$

The chain rule is applied to the partial differentiation of function in the hidden layer with respect to the hth elements of the Jacobian, resulting in

$$\frac{\partial f\left(\theta_j(t+k)\right)}{\partial u(t+h)} = \frac{\partial f\left(\theta_j(t+k)\right)}{\partial \theta_j(t+k)}\frac{\partial \theta_j(t+k)}{\partial u(t+h)} \tag{15}$$

where, the first term $\partial f\left(\theta_j(t+k)\right)/\partial\theta_j(t+k)$ is the output function's derivative, and the second term is defined by

$$\frac{\partial\theta_j(t+k)}{\partial u(t+h)} = \begin{cases} \sum_{j=1}^{m} w_{j,i+1}\delta\,(k-i,h) + \sum_{j=1}^{\min(k,n)} w_{j,n+1+i}\frac{\partial y(t+k-i)}{\partial u(t+h)}\delta\,(k-i-1) & ,k-N_u < i \\ \sum_{j=1}^{m} w_{j,i+1}\delta\,(N_u,h) + \sum_{j=1}^{n\min(k,n)} w_{j,n+1+i}\frac{\partial y(t+k-i)}{\partial u(t+h)}\delta\,(k-i-1) & ,k-N_u \ge i \end{cases} \tag{16}$$

where $\delta(h,j)$ is the Kronecker Delta function and δ_1 is the step function.

For this research, the ACC system with NNPC will have two separate controllers, upper-level and lower-level (Subiantoro et al., 2018). The upper-level part will determine the driving mode used and the reference speed. The lower level will control the amount of the gas valve and the pressure on the brakes so that the vehicle has a speed in accordance with the reference given by the upper level. The block diagram of the ACC controller with NNPC can be seen in the following figure.

Decision Algorithm

In the switching algorithm, the ACC system will select between two driving modes. When there are no obstacles or cars in front, the car's speed will change according to the value set by the driver when activating the ACC system; this mode is called cruise mode. If there is a car in front whose speed is lower than the car being driven, the driving mode will change and the ACC system will make the car's speed equal to that of the vehicle in front and maintain a safe distance. This mode is called follow mode. Data from the environment taken by vehicle sensors, such as distance between vehicles and speed of both vehicles, is used in calculation to determine the driving mode. The switching algorithm used in this paper refers to a simplified version of Gao's work (Gao et al., 2016) and can be seen in the following figure.

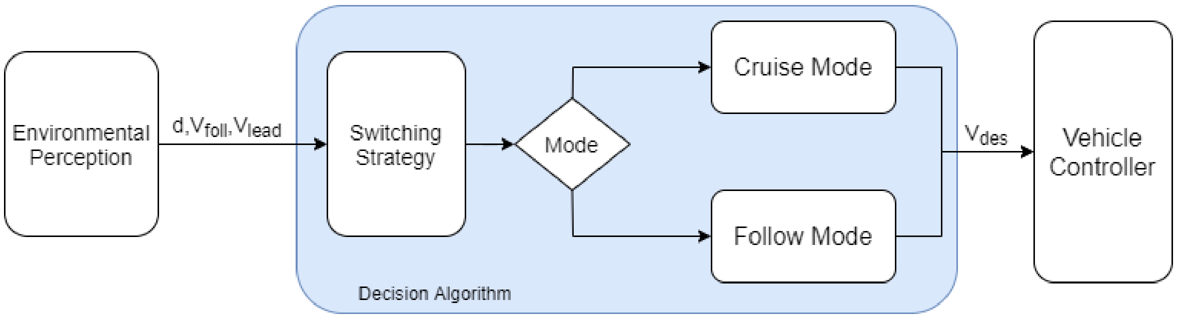


Figure 3 Switching strategy for determining cruise mode and follow mode

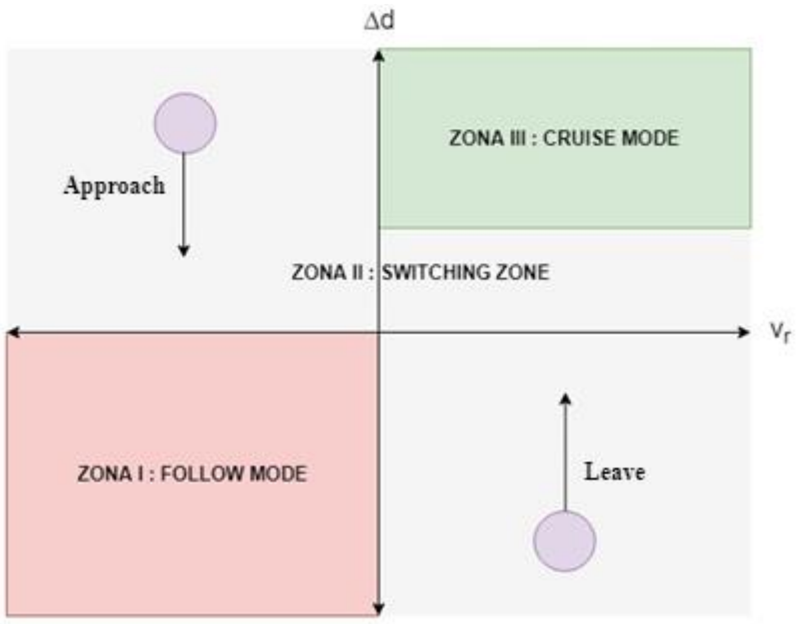


Figure 4 Switching strategy with 3 zones

In the ACC system, when there is an obstacle in the form of a vehicle with a lower speed, the system will change the driving mode through a switching algorithm. When this happens, there will be an area where the switching algorithm will change from cruise mode to follow mode and vice versa; this state is called the switching area. In the switching algorithm, there will be three zones that determine when the driving mode changes, as shown in Figure 4. Zone I is an area that has a negative relative distance and velocity. This indicates that the distance between the two vehicles is smaller than the desired safe distance, so the driving mode will change to follow mode to avoid collisions with the vehicle in front.

Zone II is the area between zones I and III. In areas that are closer to zone III, the driving mode will be more likely to be cruise mode, and in areas closer to zone I, the driving mode is more likely to be follow mode. Both driving modes can occur in zone II, making it difficult to determine the right driving mode in this zone when only looking at the two parameters used. Therefore, in this zone, an additional parameter will be considered in the form of acceleration carried out by the vehicle. This is done to smooth the switching process and avoid large changes in reference speed between cruise mode and follow mode. If this parameter does not exist, there will be a lot of switching processes so that the speed of the vehicle will oscillate greatly and make passengers uncomfortable.

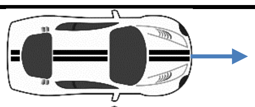
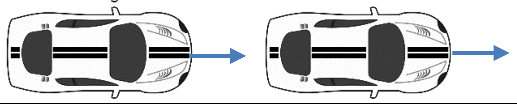
Zone III is an area where the distance between vehicles and relative speed is positive. This indicates that the distance between vehicles will get larger with the increasing speed of the vehicle in front. That way, the driving mode will become cruise mode, and the speed set by the driver will be the reference speed given to the NNPC controller. Shifting between cruise mode and follow mode can be illustrated in Table 1.

Table 1 Driving mode selection

Zone	Relative Deviation ($\Delta d = d - d_{des}$)	Relative Speed ($v_r = v_{fol} - v_{lead}$)	Acceleration Target (a)	Driving Mode
I	$\Delta d \leq d_{offset}$	$v_r \leq 0$	-	Follow Mode
II	$\Delta d > 0$	$v_r < 0$	$a_{follow} \leq a_{cruise}$	Follow Mode
	$\Delta d > 0$	$v_r < 0$	$a_{follow} > a_{cruise}$	Cruise Mode
	$\Delta d < d_{offset}$	$v_r > 0$	$a_{follow} \leq a_{cruise}$	Follow Mode
	$\Delta d < d_{offset}$	$v_r > 0$	$a_{follow} > a_{cruise}$	Cruise Mode
III	$\Delta d \geq d_{offset}$	$v_r \geq 0$	-	Cruise Mode

In the upper level controller, the PI controller will be used to determine acceleration of the follower vehicle, and then the vehicle's speed will be calculated using an ordinary linear motion function. The function to calculate acceleration in driving mode can be seen in Table 2.

Table 2 Driving mode in the switching algorithm

Mode	Movement	Acceleration
Cruise Mode		$a_{des-crs} = k_p(v_{des} - v_{fol}) + k_i \int (v_{des} - v_{fol})dt$
Follow Mode		$a_{des-fol} = k_v(v_{lead} - v_{fol}) + k_d(d - d_{des})$

Variable $a_{des-crs}$ is the acceleration needed by the vehicle to reach the target set by the driver, k_p is the proportional constant for the controller in cruise mode, v_{des} is the target speed determined by the driver when activating the ACC system, v_{foll} is the speed of the vehicle using the system ACC, k_i are integral constants in cruise mode, $a_{des-fol}$ is the acceleration needed by the vehicle when in follow mode, k_v and k_d are proportional constants to correct errors due to mismatching of the desired speed and distance. v_{lead} is the speed of the vehicle in front, d is the actual distance obtained from the sensor, and d_{des} is the desired safe distance.

3. Results and Discussion

3.1. Nonlinear Identification Results

The nonlinear identification and ACC performance are run under the CarSim simulator and MATLAB application with sampling time T_s = 0.01 seconds. Some parameters of the vehicle model are used during the simulation as follows: vehicle mass m = 1650 kg, time headway τ_h = 2, additional distance between two vehicles d_s = 1 m, and vehicle length l =

3.048 m. As a quantitative performance indicator of good fitting, a common parameter of mean square error MSE is used to verify the quality of the NNARX model.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_p - y_m)^2$$

In the first step of simulation, an NNARX model is derived by using an identification method based on data measurement. CarSim will produce 600,000 pairs of input-output data, in which only 300,000 data points (50% of output data) are used to estimate the parameter of the vehicle model, and the remaining 300,000 items (50% of output data) are used for validation. Figure 5 shows a qualitative comparison between the estimated NNARX model and the vehicle output. It is shown that the NNARX model is able to mimic the nonlinear behavior of vehicle dynamics with the value of mean square error MSE = 0.0095. The ability of the NN model as a general approximator is proved in imitating the nonlinear vehicle dynamic problem. The NN model can follow the vehicle response accurately in any situation. A small error only occurred when the vehicle speed changed suddenly.

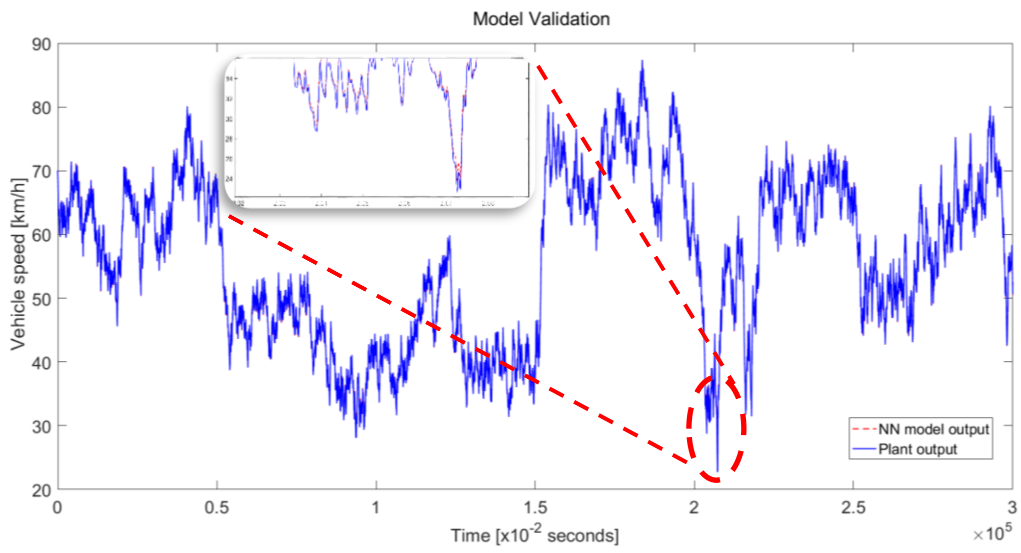


Figure 5 Comparison between NNARX model output and vehicle output

3.2. *NNPC Results*

Two different scenarios will be used as a testbed to determine controller performance. The first scenario is where the lead vehicle will go at a constant speed, then accelerate to high speed beyond the cruise mode set speed. Cruise mode speed is set at 60 km/h. For the second scenario, the lead vehicle will change in speed throughout the simulation, and cruise mode speed is set at 70 km/h. The tuning parameters of the predictive controller are determined as follows: minimum cost horizon N_1=1, prediction horizon N_2=7, control horizon N_u=2, and weighting for the control signal ρ = 0.003.

3.2.1. Control performance I: constant speed and acceleration

Figure 6 shows the simulation results for ACC mode for the first scenario. In the first 4 seconds of the simulation, the vehicle is in cruise mode with speed set to 60 km/h. Then, the distance between vehicles begins to shrink so that the speed of the follower vehicle starts to slow down to adjust to the specified safety distance and the vehicle leader's speed. At this point, ACC mode changes to follow mode until the lead vehicle starts to accelerate rapidly at the 30 second mark. The follower vehicle will adjust the speed to match the leader vehicle until it changes its drive mode to cruise mode at 60 km/h because the leader vehicle speed exceeds the specified speed set in cruise mode. The ACC system can be seen as capable of keeping a distance from the vehicle in front of it, with error from the desired distance relatively small. The largest error in distance is around 45 cm from the desired safe distance.

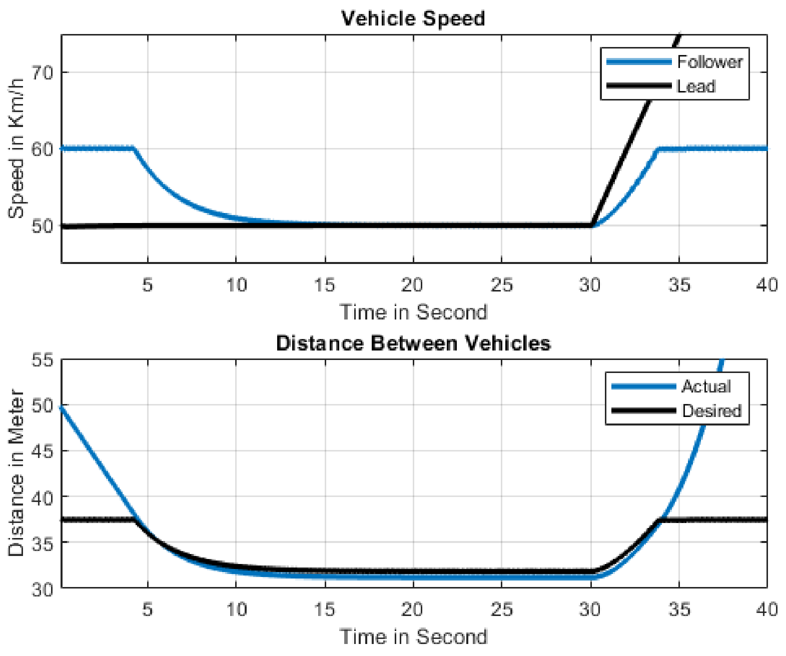


Figure 6 First scenario test

3.2.2. Control performance II: varying speed

Figure 7 shows the simulation results for ACC mode for the second scenario. At first, the vehicle is in cruise mode with speed set to 70 km/h until the distance between vehicles begins to shrink. Then drive mode changes to follow mode to adjust the leader vehicle's speed. From this point, the ACC system can keep the follower vehicle's speed to match the leader vehicle's speed within the safe distance desired. The biggest error in this scenario was around 80 cm, which happened at the 95-second mark when the leader vehicle started to decelerate.

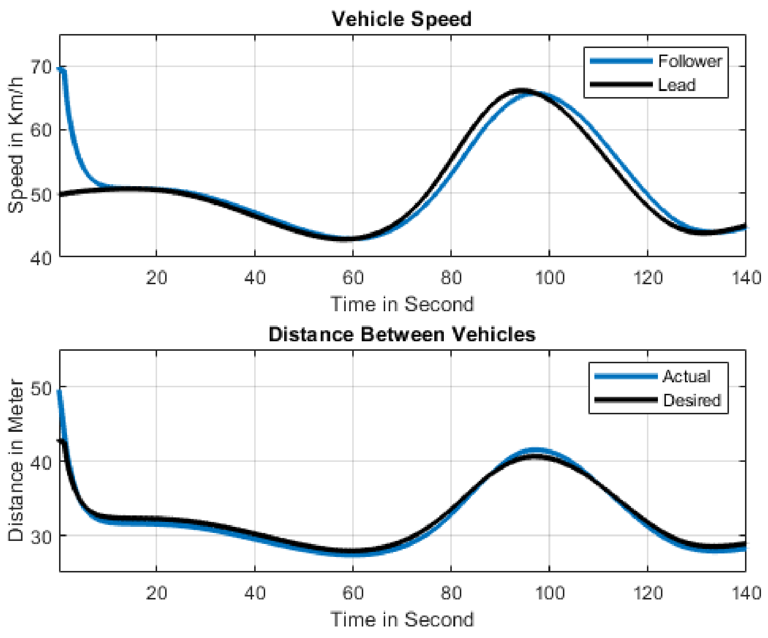


Figure 7 Second scenario test

4. Conclusions

Based on the data and simulation implemented in CarSim and MATLAB, the ACC system with NNPC can track the leader vehicle's speed and keep the safe distance desired with a relatively small error in distance. This research has limitations in acceleration of the leader vehicle. A rapid change in speed will need new training data sets for the ANN model. This method also has a drawback in the working range because the ANN model can only work accurately if there is sufficient data training, and is hard to implement in a wide working range. It would need vast data and take huge computational effort to train. For future work, another ANN model can be implemented to switch between a vehicle's speed range and acceleration to be able to work in different scenarios.

Acknowledgements

This research is funded by a Research Grant from Publikasi Terindeks Internasonal (PUTI) 2020 Universitas Indonesia.

References

ACEA, 2018. Active Vehicle Safety Most Effective, New Analysis of Accident Data Shows ACEA—European Automobile Manufacturers' Association. Available Online at https://www.acea.be/press-releases/article/active-vehicle-safety-most-effective-new-analysis-of-accident-data-shows/ Accessed on June 25, 2020

Chong, E.K.P., Zak, S.H., 2013. *An Introduction to Optimization*. 2nd Edition. New York: John Wiley & Sons, Inc

Gao, Z., Wang, J., Hu, H., Yan, W., Wang, D., Wang L., 2016. Multi-Argument Control Mode Switching Strategy for Adaptive Cruise Control System. *Procedia Engineering*, Volume 137, pp. 581–589

Martinez, J.J., Canudas-de-Wit, C., 2007. A Safe Longitudinal Control for Adaptive Cruise Control and Stop-and-Go Scenarios. *IEEE Transactions on Control Systems Technology*, Volume 15(2), pp. 246–258

Miftakhudin, M.I., Subiantoro, A, Yusivar, F., 2019. Adaptive Cruise Control by Considering Control Decision as Multistage MPC Constraints. *In*: IEEE Conference on Energy Conversion (CENCON), Yogyakarta, Indonesia, pp. 171–176

Naus, G.J.L., Ploeg, M.J.G., Van de Molengraft, W.P.M.H., Heemels, M, Steinbuch., 2010. Design and Implementation of Parameterized Adaptive Cruise Control: An Explicit Model Predictive Control Approach. *Control Engineering Practice*, Volume 18(8), pp. 882–892

Pananurak, W., Thanok, S, Parnichkun, M., 2009. Adaptive Cruise Control for an Intelligent Vehicle. *In:* IEEE International Conference on Robotics and Biomimetics, Bangkok, pp. 1794–1799

Rout, M.K., Sain, D., Swain, S.K., Mishra, S.K., 2016. PID Controller Design for Cruise Control System using Genetic Algorithm. *In:* International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), Chennai, pp. 4170–4174

Shakouri P., Ordys A., Laila D.S., 2011. Adaptive Cruise Control System: Comparing Gain-Scheduling PI and LQ Controllers. *In:* The 18th IFAC World Congress, August 28–September 2, Milano, Italy

Shakouri, P., Ordys, A., 2014. Nonlinear Model Predictive Control Approach in Design of Adaptive Cruise Control with Automated Switching to Cruise Control. *Control Engineering Practice*, Volume 26, pp. 160–177

Soloway, D., Haley, P.J., 1996. Neural Generalized Predictive Control. *In:* Proceedings of the 1996 IEEE International Symposium on Intelligent Control, IEEE

Sørensen, P.H., Nørgaard, M., Ravn, O., Poulsen, N.K., 1999. Implementation of Neural Network Based Non-Linear Predictive Control. *Neurocomputing*, Volume 28(1-3), pp. 37–51

Subiantoro, A., Fauzan, M., Feri, Y., 2018. Adaptive Cruise Control based on Multistage Predictive Control Approach. *In:* The 4th International Conference on Nano Electronics Research and Education (ICNERE)

van den Bleek, R.A.P.M., 2007. Design of a Hybrid Adaptive Cruise Control Stop-&-Go System. TNO Science & Industry Business Unit Automotive Department of Integrated Safety, Technische Universities' Eindhoven Department of Mechanical Engineering Control System Technology Group