INTERNATIONAL JOURNAL OF AUTOMOTIVE AND MECHANICAL ENGINEERING
ISSN: 2229-8649 e-ISSN: 2180-1606
VOL. 21, ISSUE 3, 2024, 11641 – 11651
DOI: https://doi.org/10.15282/ijame.21.3.2024.15.0898

RESEARCH ARTICLE

Analysis of a Simplified Predictive Function Control Formulation Using First Order Transfer Function for Adaptive Cruise Control

S.I.B. Syed Abdullah[1], M.A.S Zainuddin[1], M. Abdullah[1,2]*, K.A. Tofrowaih[3]

[1]Department of Mechanical and Aerospace Engineering, International Islamic University Malaysia, Jalan Gombak, 53100, Kuala Lumpur, Malaysia
[2]Department of Automotive Engineering Technology, Kolej Kemahiran Tinggi MARA, Masjid Tanah, 78300 Melaka, Malaysia
[3]Faculty of Mechanical Engineering Technology, Universiti Teknikal Malaysia Melaka, Hang Tuah Jaya, 76100 Durian Tunggal, Melaka, Malaysia

ABSTRACT - This paper presents a formulation and analysis of a low computation Predictive Functional Control (PFC), which is a simplified version of the more advanced Model Predictive Control (MPC) for an Adaptive Cruise Control (ACC) system by using a representation of first order closed-loop transfer function. In this work, a non-linear mathematical model of vehicle longitudinal dynamics is considered as a control plant. Then, a simple Proportional Integral (PI) controller is employed as an inner loop to identify the first-order relationship between its actual and desired trajectory speed according to the reasonable time constant based on the logical response of pedals pressing. To directly control the whole plant, the PFC is formulated as an outer loop to track the desired speed together with the convergence rate based on a user preference while satisfying constraints related to acceleration and safe distancing. Since PFC is formulated based on the first-order transfer function, the prediction and tuning processes are straightforward and specific to this system. The simulation results confirm that the proposed controller managed to track the desired speed while maintaining a comfortable driving response. Besides, the controller also can retain safe distancing during the car following application, even in the presence of unmeasured disturbance. In summary, this framework can avoid the need to formulate an inverse non-linear model that is typically used when deploying a hierarchical control structure to compute the throttle and brake pedals pressing as it has been replaced with an inner loop PI controller. The performance also is comparable yet more conservative due to the simplification. These findings can become a good reference for designing and improving the ACC controller, as the framework can be easily generalized for any type of vehicle for future work.

ARTICLE HISTORY
Received : 20th May 2024
Revised : 07th Aug. 2024
Accepted : 06th Sept. 2024
Published : 23rd Sept. 2024

KEYWORDS
Adaptive cruise control
Safe distancing
Inverse model
Model predictive control

1. INTRODUCTION

Adaptive Cruise Control (ACC) is an essential feature in autonomous vehicles for regulating speed by adjusting throttle and brake pedals according to the user's desired specifications. As computational technology has progressed, the cruise control system has transformed from its traditional form to adaptive, predictive, and even stop-and-go configurations. Nevertheless, as more sophisticated functions are embedded in the system, the algorithm's complexity is also increased, along with its computation demand [1]. Hence, it is reasonable to put extra effort into reducing computation by exploring the possibility of utilizing faster and simpler control algorithms that can provide a satisfactory response. This improvement can be valuable in the future as more computation is needed for other tasks, such as online parameter estimation for adaptive controllers [2].

In general, many control algorithms can be used for the ACC system, and the choice can vary between vehicle manufacturers and models. The traditional Proportional Integral Derivative (PID) controller is the most basic one, as it employs a feedback loop to regulate the vehicle's speed and following distance, considering current error, past error accumulation, and error rate of change. As shown in these references [3 - 5], the PID approach often delivers an acceptable control performance yet can be further improved. Nevertheless, it often struggles in complex traffic scenarios due to the selection of global gains to represent a vehicle's entire non-linear dynamics. Indeed, several modification approaches can be used, such as gain scheduling [6], integration with Fuzzy logic [7], adaptive correction with Kalman filter [8], and a special tuning algorithm [9]. Yet the overall tuning and implementation are not that transparent and intuitive, requiring complex alteration and modification of the existing algorithm.

Fuzzy logic controllers are another option that can be used for the ACC system. It employs linguistic variables to handle uncertain or vague inputs. Thus, it can effectively manage non-linear and complex traffic scenarios. By providing adaptive responses, the gap between traditional control methods and more advanced approaches can be reduced. For example, Basjaruddin and his team have shown that the ACC system can provide reliable performance in different driving scenarios [10]. Despite its successful implementation, a user should note that a customized rule set is needed and often derived based on a trial-and-error process. Besides, most of the rules are system-dependent, which is difficult to generalize for other types of vehicles. This issue can be overcome with the help of neural network methods, such as the deep learning

*CORRESPONDING AUTHOR | M. Abdullah | ✉ mohd_abdl@iium.edu.my

method that enables ACC systems to compute a control action based on vast amounts of sensor data. Neural networks can learn from historical data and adapt to diverse driving conditions and different types of vehicles, as shown in these references [11,12]. However, their effectiveness often hinges on the quality and quantity of data along with their tedious training and validating processes.

Model Predictive Control (MPC) is another favored method for the ACC system, where it takes a predictive approach to compute optimal control action at every time step with the help of a quadratic cost function. By utilizing the internal mathematical model representation of vehicle dynamics, MPC can anticipate the future behavior of both the host and lead vehicles. As shown in the work of [13,14], by considering multiple factors such as speed, acceleration, and road conditions, MPC can optimize the host vehicle's speed while ensuring a safe following distance and other sets of driving constraints such as fuel efficiency. Although MPC excels in handling intricate traffic situations, it requires substantial computational resources, which is the primary concern of car developers. Indeed, various methods exist to improve, such as offline implementation [15], representing the finite prediction with the Laguerre function [16] and deep reinforced learning [1], yet the computation is still considered heavy, especially when constraints are imposed.

Another suitable option for the ACC system is to use Predictive Functional Control (PFC), the simplified version of MPC with faster computation and more straightforward formulation yet a bit suboptimal. Instead of using the quadratic cost function, PFC computes the control action based on the minimization between predicted output and first-order target trajectory in a specific coincidence point. The main tuning parameters are the Closed-loop Time Response (CLTR), which is the time needed to reach 95% from the steady state value, and the coincidence horizon itself. By using these values, the tuning process becomes more intuitive and straightforward. Several works have used PFC for the ACC system, such as references [5,17], where it is shown that the PFC controller is quite capable when benchmarked with MPC and PID controllers. Nevertheless, one issue with PFC is when the representative model is not a first-order transfer function, the coincidence horizon must be selected carefully. As reported by Rossiter and Abdullah [18], a too-small horizon will lead to aggressive actuation as it forces the system to follow precisely the first-order target trajectory. In contrast, the large horizon will reduce the efficacy of CLTR as the tuning parameters. Indeed, there are also several solutions or improvements to overcome the previous issue with PFC over these past years, such as integrating it with exponential decay prediction [19] and adjusting the target trajectory [18]. Yet, the theory is quite heavy, and there is no guarantee that it will work except for the first-order dynamic internal model [18].

Note that when designing a controller for an ACC system, a hierarchical control structure is often used where it consists of cascade loops. The outer loop typically utilizes the main controller, such as PFC [17] or MPC [14]. In contrast, the inner loop usually consists of an inverse non-linear longitudinal model that maps the output signal from the outer loop controller to the actual actuation in a vehicle: the pedal and brake pressing. Many references adopted this structure as it can simplify the design process since the outer loop is derived based on the linear kinematic relationship. Nevertheless, deriving an accurate inverse model is a challenging task. If a first principal model is derived, a user must find all the required vehicle parameters, such as the moment of inertia, mass, cross-section area of a vehicle, and others. Similarly, an alternative system identification method requires experimental data where the signal must be adequately designed to capture the essential system dynamics.

Based on the literature survey, PFC seems a good option. However, if the representative model is other than the first-order transfer function, the selection of coincidence horizon may affect the overall performance. The second issue is related to the inverse model, where either system identification or the first principal model is required to get the relationship between the outer loop output signal and the actual actuation of a vehicle. Hence, in this work, the main contribution is to analyze the positive and negative effects of simplifying the inverse model by using PI or PID controller as the inner loop to represent the overall closed-loop dynamic of the plant as a first-order transfer function. With this implementation, the formulation of outer-loop PFC, including its tuning and constraint handling algorithm, will become more transparent and straightforward, while the effect of first-order simplification will depend on the robust nature of the PFC algorithm itself.

2. METHODOLOGY

This section starts with a brief description of vehicle longitudinal dynamics that will be used as a plant to represent a real car. The following subsection will discuss the inner loop structure of a simple PI controller to get the first-order representative transfer function. Then, the outer-loop controller is derived using PFC formulation, including its constraints handling approach. The final subsection will provide the methodology to set up a car following the application to test the proposed controller.

2.1 Vehicle Plant

In this work, a vehicle longitudinal dynamic consisting of the power train and braking systems is used from the work of Syed et al. [20]. Since a detailed description of the model has been presented in the paper, only a general mathematical expression will be presented in this work. Figure 1 shows the block diagram of the plant where the input to the system is pedal pressing from the brake x_b and throttle x_t, while the output is velocity v. The input from the throttle pedal will generate the tractive force F_t, while the input from brake pressing will generate braking force F_b. The summation of tractive and braking force will contribute to vehicle speed calculation.

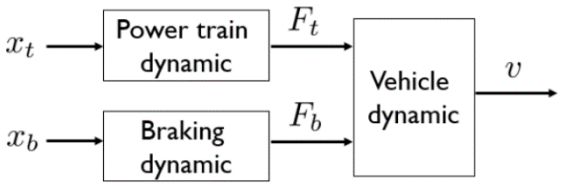

Figure 1. Block diagram for vehicle plant [20]

The simplified equation for this system can be represented as:

$$v = f(F_t, F_b) \tag{1}$$

where F_t and F_b are:

$$F_t = f(x_t)\,;\, F_b = f(x_b) \tag{2}$$

For more detailed parameters and equations, an interested reader can refer to this reference [20]. It should be noted that this system is non-linear, and thus, implementing a linear controller will lead to some inaccuracy in a particular operating point. As discussed in the introduction, a hierarchical control structure is often employed where the upper controller is derived based on a linear kinematic model. At the same time, the lower level will track the produced signal by using an inverse non-linear longitudinal model. Nevertheless, obtaining the inverse model is also quite challenging, even with a system identification approach where there will be some uncertainty that can affect the tracking process of the lower-level system.

2.2 Inner Control using PI Controller

To replace the inverse model equation, a simple Proportional Integral (PI) controller is used to estimate the relationship between the inner-loop input u and the actual velocity v. With this method, a designer does not need to consider the overall dynamics of a vehicle. Figure 2 shows the schematic of the inner loop control using a PI controller, where the gain will be tuned directly from the plant by trial and error without using a mathematical model.

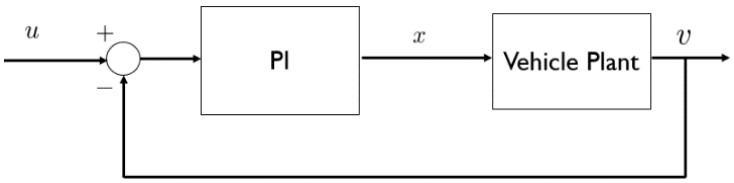

Figure 2. Block diagram for the inner loop

Nevertheless, in the actual implementation, the tuning process should be conducted carefully to avoid hardware damage. For safety reasons, a designer should start from a small proportional gain k_p and gradually increase it to minimize the steady-state error as much as possible. Similarly, to eliminate the steady-state error, the integral gain k_i should start with a small value and gradually increase it to a suitable one. The outcome should not possess any overshoot with an acceptable time constant or convergence rate. Once a satisfactory response is obtained, a first-order transfer function can be identified based on the obtained closed-loop response from Figure 2 to represent the overall inner loop response as:

$$v = \frac{1}{\tau s + 1} u \tag{3}$$

where τ is the estimated time constant from the closed-loop PI response, thus, the transfer function can be used to formulate the PFC algorithm for the outer-loop controller.

Remark 2.1: For the inner loop, it is also possible to use a full PID controller instead of a PI controller, depending on the vehicles used. The relationship is determined based on the basis of producing a pre-determined closed-loop response that gives zero steady-state error and minimum overshoot so that it can be estimated or identified easily using the first-order model as in Equation (3). Since in this work, the PI controller alone can produce a satisfactory response for the identification process, it will be adopted instead of PID controller.

2.3 Outer Control using PFC Controller

The overall block diagram for the PFC control law is presented in Figure 3, where R is the desired velocity. Since the inner loop, which refers to the black dashed box, is already presented as a first-order transfer function as in Equation (3), the PFC algorithm will be formulated based on it.

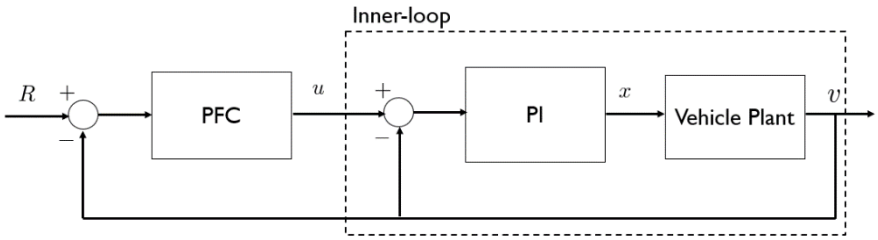

Figure 3. Block diagram for PFC-PI controller

The PFC formulation is derived based on a discrete time domain, and the transfer function in Equation (3) is discretized with a suitable sampling time (for this case, T_s = 0.01s). This step can be done by utilizing the *c2d* function in MATLAB. The discrete transfer function can be represented as:

$$v(z) = \frac{bz^{-1}}{1 - az^{-1}}u(z) \tag{4}$$

where a and b are the respective transfer function parameters, the parameter v represents the velocity that the model calculates, while z^{-1} is the shift operator. Rearranging Equation (4):

$$[1 - az^{-1}]v(z) = [bz^{-1}]u(z) \tag{5}$$

Taking note of the shift operator z^{-1}, Equation (5) can be formulated in a sample form k as:

$$v(k) - av(k-1) = bu(k-1) \tag{6}$$

where k is the value of the current sample and $(k-1)$ is the value of the previous sample. Applying the superposition principle, the one-step-ahead prediction of the model at the current sample can be represented as:

$$v(k+1|k) = bu(k) + av(k) \tag{7}$$

To get an unbiased prediction, the term $d(k) = v_p(k) - v(k)$, which is different between the actual plant output and model output, is added in Equation (7) and thus:

$$v(k+1|k) = bu(k) + av(k) + d(k) \tag{8}$$

For the PFC algorithm, a first-order target trajectory is often employed due to the ease of tuning and implementation, although other sorts of trajectory can also be implemented [21]. The one-step-ahead trajectory can be represented as:

$$v(k+1) = R - \Big(R - v_p(k)\Big)\lambda \tag{9}$$

where λ is the desired pole or, in other terms, it represents the desired Closed Loop Time Response (CLTR) with the relation of:

$$\lambda = e^{T_s/CLTR} \tag{10}$$

where T_s is again the sampling time. This parameter represents the time required to achieve 95% from the steady state value, which in turn becomes the main tuning parameter of PFC.

Since the internal model of PFC is a first-order transfer function, according to many references [18,21], only one step ahead of prediction is required to compute the control action to avoid ill-posed solutions and better tracking performance. The choice of prediction will be different should a higher-order transfer function be used. The control law of PFC can be formulated by equating the one-step-ahead prediction as in Equation (8) with the set point trajectory of Equation (9) while solving for the input as:

$$u(k) = \frac{\Big[R - \Big(R - v_p(k)\Big)\lambda\Big] - [av(k) + d(k)]}{b} \tag{11}$$

2.4 Acceleration Constraints

When driving a car, it is essential to take note of the acceleration and deceleration as they can affect the comfortability of the passenger and fuel consumption. Typically, the comfort level of acceleration is defined from the range of $a_{min} = -3$ m/s^2 to $a_{max} = 2$ m/s^2 [14,22]. Hence, the ACC system should limit its acceleration within this range. Ideally, the inner model of PFC can be used to predict the future acceleration based on its current computed input such that:

$$a(k+1|k) = \frac{v(k+1|k) - v(k)}{T_s} \tag{12}$$

If $a(k+1|k) < a_{min}$ or $a(k+1|k) > a_{max}$, then the new input $u(k)$ can be computed by noting the future velocity in Equation (8) as:

$$u(k) = \frac{a_{max/min}T_s + v(k) - av(k)}{b} \tag{13}$$

Since the acceleration dynamic in PFC is designed to be monatomic, then only the one-step-ahead prediction is needed. Nevertheless, if there is a slight plant-model mismatch, the algorithm may end up not detecting any constraint violation, in addition to the fact that acceleration is highly dependent on the value of velocity.

This issue can be considered as one of the weaknesses of this algorithm, where the acceleration constraint is not suitable to be treated as an output constraint. A simple solution is to translate the limit into the rate input constraints since u is the velocity reference signal for the inner PI controller (similar to the reference governors' approach). In case the system meets the boundary condition, it will be satisfied by using the rate input constraints method. The algorithm below demonstrates how the constraint is computed as:

if $u(k) - u(k-1) > a_{max}T_s$
then
compute $u(k) = a_{max}T_s + u(k-1)$
end

if $u(k) - u(k-1) < a_{mim}T_s$
then
compute $u(k) = a_{min}T_s + u(k-1)$
end

2.5 Relative Distance Constraint

For a vehicle following application, an ego vehicle (controlled vehicle) must maintain a safe following distance regardless of any specified desired speed. This requirement ensures safety and avoids any possible collusion with a lead vehicle. In general, there are many methods to calculate a safe distance ranging from a constant value to a varying value that depends on the speed of the lead vehicle. Based on reference [22], the safe distance prediction can be computed as:

$$D_{safe} = D_{default} + T_{gap}v(k) \tag{14}$$

where $D_{default}$ is the default value of safe distance in a complete stop, it is typically assumed as 10 m. The parameter T_{gap} represents a safe time gap often set to 1.4 s [22].

Figure 4. Block diagram for vehicle following application

Figure 4 shows the overall signal feedback for the vehicle following the application. The PFC-PI controller block represents the whole cascade block, as shown in Figure 3, which consists of the PFC controller, inner PI loop and vehicle mathematical model. To implement the safe distance constraint, the controller needs two important values: the relative velocity v_r and relative distance D_r that represents the difference between lead and ego vehicles. The future relative distance can be predicted based on the current value of the two parameters such that:

$$D\_r(k+1|k) = [v_l - v(k)]T_s + D_r(k) \tag{15}$$

Assuming that the lead vehicle velocity is constant and $D_r(k+1|k) > d_{safe}$, the safe distance can be converted to output velocity constraints by equating Equation. (14) and Equation (15) to get the maximum allowable velocity at each prediction that can be used to check the violation as:

$$v_{max}(k+1) = \frac{v_l T_s + D_r(k+n-1) - D_{default}}{T_{gap} + T_s} \tag{16}$$

From Equation (8), the n step ahead future velocity prediction can be formulated in a matrix form as:

$$V = HU + Fv(k) + Ld(k) \tag{17}$$

where:

$$V = \begin{bmatrix} v(k+1|k) \\ v(k+2|k) \\ \vdots \\ v(k+n|k) \end{bmatrix}; H = \begin{bmatrix} b & 0 & 0 & 0 \\ ab & b & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots \\ a^{n-1}b & a^{n-2}b & \cdots & b \end{bmatrix}; U = \begin{bmatrix} u(k) \\ u(k+1) \\ \vdots \\ u(k+n-1) \end{bmatrix}; F = \begin{bmatrix} a \\ a^2 \\ \vdots \\ a^n \end{bmatrix}; L = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$$

For simplicity of computation, the future input U is assumed to be constant where $u(k) = u(k+1) \ldots = u(k+n)$ [21]. Thus, the prediction equation is simplified to:

$$V = HLu(k) + Fv(k) + Ld(k) \tag{18}$$

where for accessing a specific prediction sample, only the respective row in the matrix is used as:

$$v(k+n|k) = H_n Lu(k) + F_n v(k) + L_n d(k) \tag{19}$$

If the future predicted velocity of the ego vehicle at the current sample in Equation (19) is more than $v_{max}(k+n)$, then new control input is computed as:

$$u(k) = \frac{v_{max}(k+n) - [F_n v(k) + L_n d(k)]}{H_n L} \tag{20}$$

It is important to consider more than one step ahead of constraint checking (validation horizon) at each sampling time to avoid aggressive acceleration and deceleration when satisfying the limit. Thus, the value for $D_r(k+n)$ need to be updated accordingly with the ego vehicle velocity prediction as in Equation (19). For clarity, the algorithm below explains the complete loop on how to implement the constraints:

At each sample, compute v_{max} as in Equation (16)

for i = 1:n

compute $v(k+i)$ as in Equation (20) and compute $v_{max}(k+i)$ as in Equation (16)

if $v(k+i) > v_{max}$

compute $u(k)$ as in Equation (20)

update $v(k+i)$ in Equation (19) with new value of $u(k)$

end

update $D_r(k+i)$ as in Equation (15)

end

Remark 2.2: Since the acceleration and safe distance constraints are computed separately, it is crucial to program them sequentially according to their importance. As safe distance is related to safety, it needs to be implemented later after implementing the acceleration constraints, which are more related to passenger comfort.

3. SIMULATION RESULT

This section presents the simulation results for the proposed controller. The following two subsections will provide the impact of tuning for the inner-loop using the PI controller and outer-loop performance of PFC in unconstrained and constrained environments, respectively. The final subsection demonstrates the controller's capability in a car following the scenario as compared to a benchmarked PFC with the inverse model to analyze its performance in terms of safe distancing and disturbance rejection.

3.1 Inner Loop using PI Controller

The first step in designing a hierarchical control structure is tuning the inner-loop controller. As discussed previously, this part often consists of an inverse non-linear model to compute the plant input by referring to the supplied outer-loop signal. Nevertheless, in this work, a simple PI controller is employed to overcome the challenges in modeling the non-linear longitudinal dynamic of a vehicle. Figure 5 shows the tuning effect of the PI controller and its control effort. Given that the PI gains are tuned by using the trial-and-error method. Nevertheless, when tuning the PI controller, one should also note the control effort when selecting the gains so that the actual input is between 100% (representing maximum throttle pressing) and -100% (representing maximum brake pressing). The selected three gains combinations are presented as:

- PI 1: $k_p = 8$, $k_i = 3$
- PI 2: $k_p = 4$, $k_i = 0.15$
- PI 1: $k_p = 2$, $k_i = 0.1$

The set point is designed to track the usual speed limit such that for urban areas (10 m/s), single carriage roads (16 m/s), dual carriage roads (25 m/s), and highways (30 m/s). This profile does not represent the drive cycle analysis but

rather assesses the tuning efficacy of PFC during steady state change to analyze how well the controller retains the selected desired CLTR in every change, as the default mode of any cruise control is to track a constant speed rather than variable speed. The main reason why drive cycle analysis is not used is because it is designed to simulate a wide range of driving conditions. They include various phases, such as acceleration, deceleration, idling, and constant-speed driving, reflecting urban, suburban, and highway scenarios. The purpose of these cycles is to assess overall vehicle performance, including fuel efficiency, emissions, and energy consumption, under standardized conditions. At the same time, ACC systems are designed to maintain a specified distance from the vehicle ahead, adjusting speed as necessary. This critical aspect of ACC functionality cannot be assessed using drive cycles, which do not involve following other vehicles or managing distances.

Based on Figure 5, it should be noted that PI 1 (blue dotted line) provides the fastest response, followed by PI 2 (red solid line) and PI 3 (green dashed line). This finding aligns with the controller's root mean square error (RMS) compared to the setpoint, as presented in Table 1. Nevertheless, a user also needs to consider the control effort of the pedal pressing, where PI 1 requires more aggressive actuation and complete brake pushing when the desired speed is dropped. This behavior must be avoided since the outer-loop control does not directly consider the vehicle's physical constraints. Conversely, PI 3 provides a slower response, although the control effort is the least aggressive. In this case, PI 2 is the most suitable to select as its time constant is around 5s, although it is not for every change in the set point due to non-linearity in the vehicle plant.

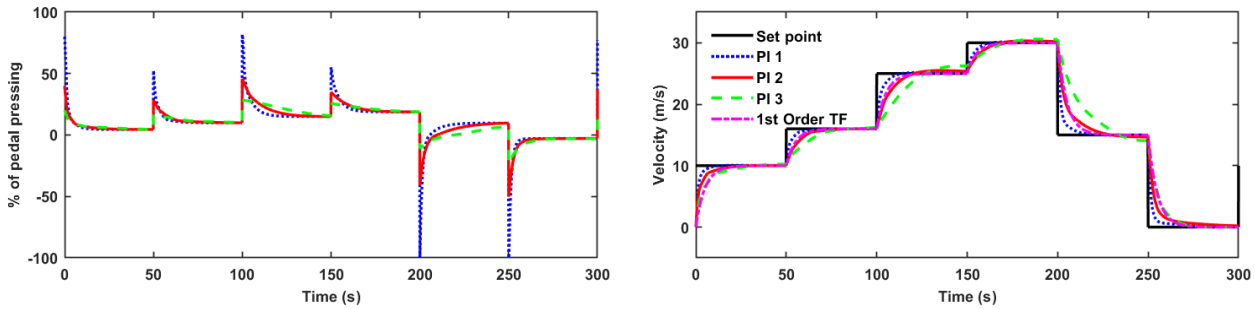

Figure 5. Inner-loop PI controller performance with different gains combination

Since PI 2 is considered the inner-loop controller, a first-order transfer function with a time constant of 5s can be used to represent the overall loop for PFC formulation. In the same figure, the actual velocity produced by PI 2 is the closest to the 1st-order transfer function response (magenta dash-dotted line) when given the same desired speed as the other two PI controllers. For qualitative measurement, Table 1 provides the root mean square error between the three PI controllers and the first-order transfer function response, whereas PI 2 gives the lowest error with a value of 0.3642.

Table 1. RMS of the response as compared to the set point and transfer function

Controller	RMS compared to set point	RMS compared to transfer function
PI 1	2.2414	1.5579
PI 2	4.9006	0.3642
PI 3	9.8225	1.6685

3.2 Outer Loop using PFC Controller

The next step is to deploy the PFC algorithm for the outer loop based on the representative first-order transfer function obtained from the previous section. The outer controller is responsible for providing the reference signal to the inner loop so that users can change the CLTR according to their desired specifications while satisfying the output constraints. In this analysis, the interest is to examine how well the closed-loop response provides the desired CLTR. Figure 6 shows the unconstrained closed-loop performance of the PFC-PI controller with different selections of CLTR value. As can be observed, the proposed controller managed to track the desired velocity profile. However, as previously discussed, there is a slight plant model mismatch in the representative first-order transfer function. Besides, the PFC's tuning parameter significantly impacts the overall performance. With CLTR of 5s (blue-dotted line), 10s (red dashed line), and 15s (green solid line), the velocity produced reaches 95% from the steady state value, which closely matches the selected CLTR with slight variation in each setpoint changes as given in Table 2. This is a good feature, especially when driving in a different condition. For example, faster CLTR may be desirable during low-speed operation to avoid phantom traffic, while slower CLTR is desired for comfortability during highway mode. Thus, in the future, adaptive scheduling of vehicle response that depends on operation speed can be employed by using this algorithm.

Table 2. Data analysis for tuning efficacy of outer PFC controller

Desired CLTR	Variation in actual CLTR
5s	±0.6s
10s	±3.7s
15s	±4.5s

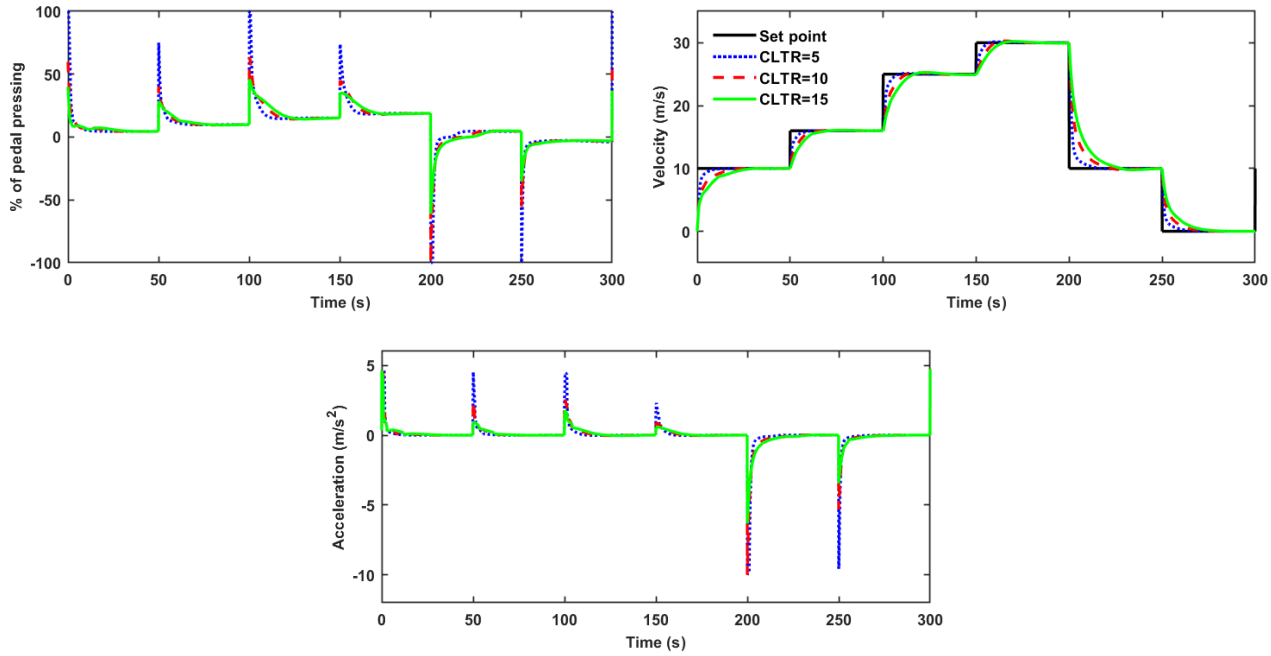

Figure 6. PFC-PI controller response with different CLTR

Nevertheless, in this work, only one CLTR will be used for the whole operation. Based on Figure 6, using CLTR=5 will require a high control effort since the vehicle needs to reach the set point in a short amount of time. As seen from the pedal pressing, both throttle and brake pedals must be fully pressed in certain situations. Thus, CLTR=15 will be used throughout this work since this value can be generalized for the whole operation. However, when referring to the acceleration graph, the vehicle produced acceleration and deceleration beyond 2 m/s^2 and -3 m/s^2, respectively. This will provide an uncomfortable ride for the passenger; thus, output constraints must be implemented.

3.3 Analysis of the Acceleration Constraint Implementation

In this section, the comparison for constrained cases is analyzed based on the violation of the boundary conditions. For the acceleration constraint, ideally, the internal model is used to predict whether the acceleration will violate the constraints. This can be done by treating the acceleration as an output constraint. Nevertheless, as can be observed from Figure 7, the controller fails to prevent the violation due to the plant model mismatch since the acceleration is highly sensitive to a small change in velocity. Thus, even a slight mismatch will lead to a failure to detect the violation, or it will be too late to impose any changes to satisfy the constraints. This can be identified as one of the weaknesses of this control structure.

Alternatively, as discussed in the methodology section, instead of treating acceleration as an output constraint, it can also be translated to the rate input constraint since the input is a trajectory reference for the inner PI controller, as in a reference governor approach. The response can be seen in Figure 7, where most of the acceleration satisfies the limits except in certain regions. Nevertheless, the resulting performance has become too conservative, as seen from the acceleration response. There is a gap between the constraint line and the actual response. For future improvement, an alternative model representation may be desirable such that multiple first-order models to identify the plant dynamics. For this control structure, one should expect the weaknesses considering the algorithm's simplicity. Table 3 summarizes the observation based on the result presented in Figure 7.

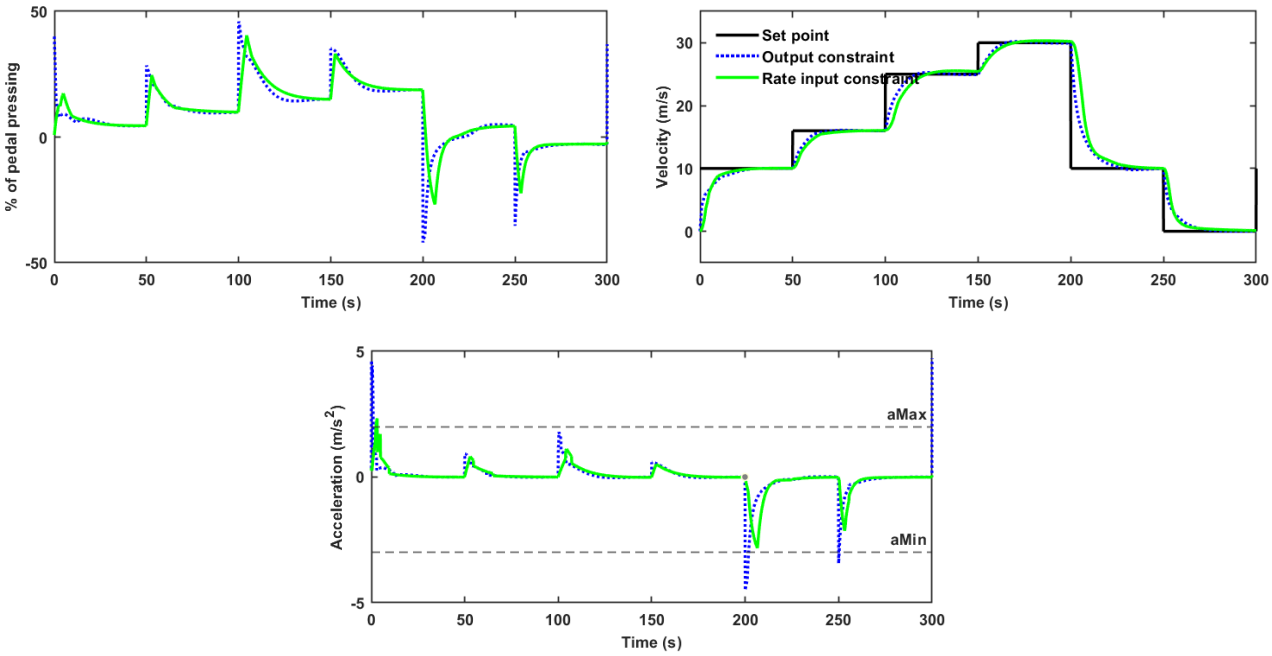

Figure 7. Constrained PFC-PI controller response with different validation horizon n_i

Table 3. Constrained performance when boundary conditions are met

Method	Violation of limit
Set of output constraints	Minimum violation at 1s
Set as rate input constraints	Violation at 1s and 200s

3.4 Car Following Scenario

To further evaluate the performance of the proposed algorithm, a car following scenario is considered where a lead vehicle will change its speed according to the sinusoidal wave between 25 to 31 m/s to mimic the common driving behavior. In this case, the controller is set to track the desired speed of 30 m/s while considering an additional safe distancing constraint. Figure 8 shows the velocity, acceleration, relative distance, and percentage of pedal pressing responses for the PFC-PI controller as compared to the benchmarked PFC with an inverse model (CPFC-Inv).

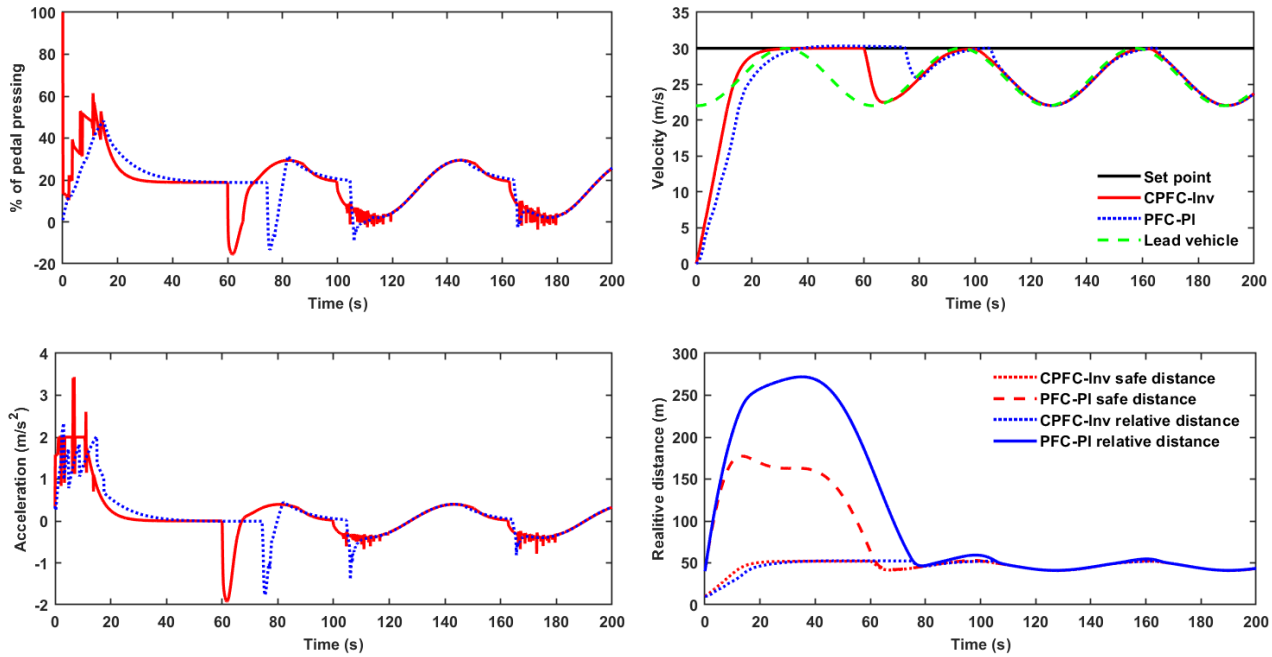

Figure 8. Comparison between CPFC-inv and PFC-PI for vehicle following application

For this analysis, the interest is on how well the controller reacts to satisfy the safe distancing requirement. In the velocity graph, the green dashed line represents the speed of the lead vehicle. Both ego cars, implemented with a controller, start at 25 m/s and increase their speed to reach the desired set point (black dashed line). For PFC-PI (blue

dotted line), the controller detected the possibility of the car violating the safe distance constraint around 65 s and automatically reducing its speed. Once the lead vehicle increases its speed, the ego vehicle continues to track the desired speed until 100s, and after that, the same scenario is repeated. Throughout the simulation, the acceleration of the ego vehicle is kept within the limit of -3 to 2 m/s^2 except in certain situations due to the plant model mismatch, and the percentage of pedal pressing is also not that aggressive. Referring to the CPFC-inv (red solid line), the output is more responsive as compared to the PFC-PI. Additionally, it also follows the lead vehicle velocity profile more smoothly with abrupt changes in acceleration. This is because, as compared to the PFC-PI, it detects violations a little bit late, and as a result, more changes need to be applied to the acceleration. Nevertheless, there is some fluctuation occurred due to the plant model mismatched.

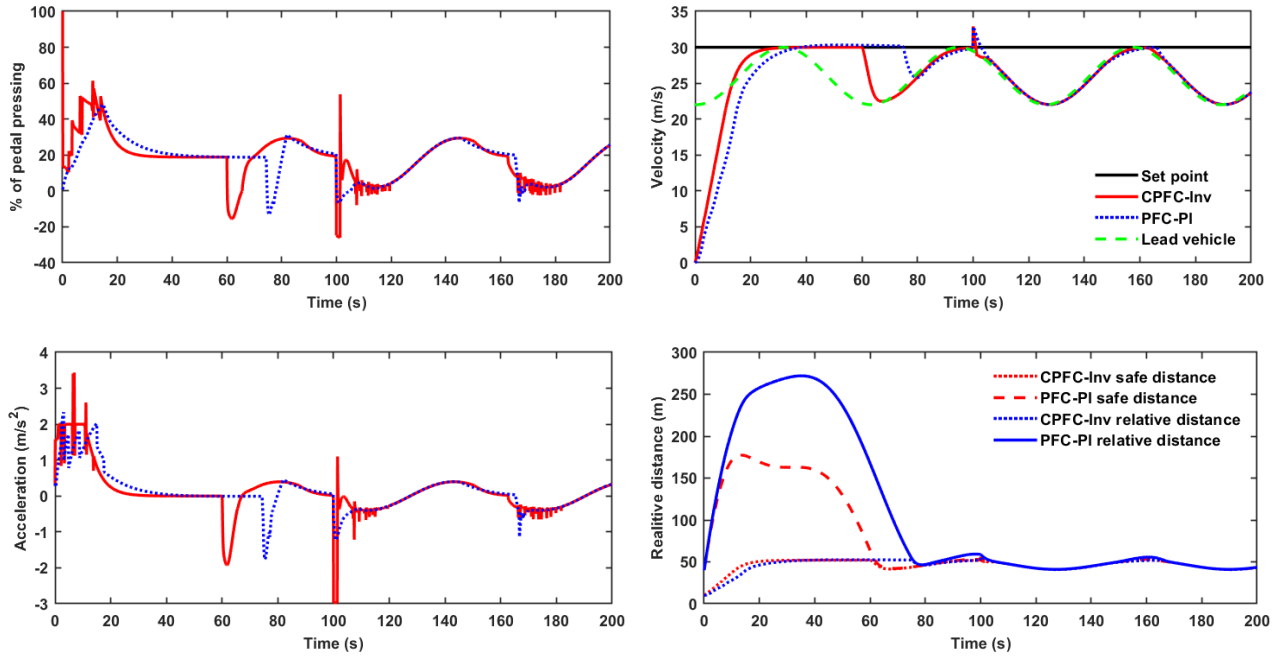

Figure 9. Comparison between CPFC-inv and PFC-PI for positive disturbance

The same situation can be observed when a small disturbance is added to the output that going up or down the hill. Figure 9 shows the response when a continuous disturbance output of 2 m/s is added at 100s interval time in the simulation. As can be seen, when the speed suddenly increases, the safe distance also decreases. Thus, the vehicle needs to decelerate at approximately -3 m/s^2 to comply with the safe distancing. Apart from that, the controller resumed normal operation in tracking the set point while ensuring safe distancing from the lead vehicle. When compared to the benchmarked controller, CPFC-inv provides a more aggressive response in the declaration due to its higher speed when the disturbance is inserted, yet the corrective action is faster than the PFC-PI controller. Table 4 shows the comparison between PFC-PI and CPFC-inv in terms of the first response time of the speed change and the response time taken to reject the disturbance. Although the performance is different, both controllers managed to track the setpoint while retaining a safe distance even in the presence of disturbance.

Table 4. Response time comparison between PFC-PI and CPFC-inv

Controller	First reaction time for speed change	Response time to reject disturbance
PFC-PI	65s	8s
CPFC-inv	80s	11s

4. CONCLUSION

In summary, this work has proposed, formulated, and simulated the hierarchical PFC-PI control algorithm specifically for the ACC system. Using the PI controller for the inner loop, a user can avoid using an inverse model to compute the required amount of pedal pressing based on the outer-loop signal. The second advantage is that a first-order transfer function can be identified and used as the representative model to formulate the control law of PFC. This feature makes the tuning process of ACC straightforward as, by default, the choice of coincidence horizon is always one. At the same time, a user can easily change the desired response based on CLTR, which is handy should adaptive ACC need to be implemented based on different driving scenarios in the future. The results also show that the proposed controller managed to produce an acceptable response despite unmeasured disturbance.

Nevertheless, as expected, the simplicity of the first-order transfer function limits the ability to satisfy the acceleration constraints. Thus, instead of formulating it as an output constraint, it is formulated as a rate constraint to mimic the

reference governor approach. As a result, the response is quite conservative as compared to the benchmark controller that used an inverse model. For future work, different simple model structures can be explored to overcome this weakness.

ACKNOWLEDGEMENTS

This work was funded by a grant from the Ministry of Higher Education of Malaysia under Fundamental Research Grant FRGS/1/2021/TK02/UIAM/02/2 (FRGS21-240-0849).

CONFLICT OF INTEREST

The authors declare no conflicts of interest.

REFERENCES

[1] Y. Lin, J. McPhee, and N.L. Azad, "Comparison of deep reinforcement learning and model predictive control for adaptive cruise control," *IEEE Transactions on Intelligent Vehicles*, vol. 6, no. 2, pp. 221-231, 2020.

[2] S. Liu, L. Liu, J. Tang, B. Yu, Y. Wang, and W. Shi, "Edge computing for autonomous driving: Opportunities and challenges," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1697-1716, 2019.

[3] S. Chaturvedi and N. Kumar, "Design and implementation of an optimized PID controller for the adaptive cruise control system," *IETE Journal of Research*, vol. 69, no. 10, pp.7084-7091, 2023.

[4] S. Chamraz and R. Balogh, "Two approaches to the adaptive cruise control (acc) design," in *2018 Cybernetics & Informatics (K&I)*, IEEE, pp. 1-6, 2018.

[5] M. Zainuddin, M. Abdullah, S. Ahmad and K. Tofrowaih, "Performance comparison between predictive functional control and pid algorithms for automobile cruise control system," *International Journal of Automotive and Mechanical Engineering*, vol. 19, no. 1, pp. 9460-9468, 2022.

[6] P. Shakouri, A. Ordys, D.S. Laila and M. Askari, "Adaptive cruise control system: Comparing gain scheduling PI and LQ controllers," *IFAC Proceedings*, vol. 44, no. 1, pp. 12964-12969, 2011.

[7] G. Prabhakar, S. Selvaperumal and P.N. Pugazhenthi, "Fuzzy PD plus I control-based adaptive cruise control system in simulation and real-time environment," *IETE Journal of Research*, vol. 65, no. 1, pp. 69-79, 2019.

[8] F. Islam, M.M. Nabi, M.M. Farhad, P. Peranich, J.E. Ball, and C. Goodin, "Evaluating performance of extended Kalman filter based adaptive cruise control using PID controller," in *Autonomous Systems: Sensors, Processing, and Security for Vehicles and Infrastructure 2021*, SPIE, vol. 11748, pp. 46-56, 2021.

[9] A. Laith, E. Serdar, I. Davut and Z.R. Abu, "Modified elite opposition-based artificial hummingbird algorithm for designing for PID controlled cruise control system," *Intelligent Automation & Soft Computing*, vol 38, no. 2, 169-183, 2023.

[10] N.C. Basjaruddin, K. Kuspriyanto, D. Saefudin and I.K. Nugraha, "Developing adaptive cruise control based on fuzzy logic using hardware simulation," *International Journal of Electrical and Computer Engineering*, vol. 4, no. 6, p. 944, 2014.

[11] P. Mahadika, A. Subiantoro and B. Kusumoputro, "Neural network predictive control approach design for adaptive cruise control," *International Journal of Technology*, vol. 11, p. 1451, 2020.

[12] Y.C. Lin and H.L.T. Nguyen, "Adaptive neuro fuzzy predictor-based control for cooperative adaptive cruise control system," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 1054-1063, 2019.

[13] R. Zhao, P.K. Wong, Z. Xie and J. Zhao, "Real-time weighted multi-objective model predictive controller for adaptive cruise control systems," *International Journal of Automotive Technology*, vol. 18, pp. 279-292, 2017.

[14] T. Takahama and D. Akasaka, "Model predictive control approach to design practical adaptive cruise control for traffic jam," *International Journal of Automotive Engineering*, vol. 9, no. 3, pp. 99-104, 2018.

[15] A. Weißmann, D. G¨orges and X. Lin, "Energy-optimal adaptive cruise control based on model predictive control," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 12563-12568, 2017.

[16] U. Karapınar and L. G¨oren-S¨umer, "Laguerre MPC formulation for automotive adaptive cruise control application and performance analysis," in *2018 6th International Conference on Control Engineering & Information Technology*, pp. 1-6, 2018.

[17] M.A.S. Zainuddin, M. Abdullah, S. Ahmad, M.S. Uzir and Z.M.P.A. Baidowi, "Performance analysis of predictive functional control for automobile adaptive cruise control system," *IIUM Engineering Journal*, vol. 24, no. 1, pp. 213-225, 2021.

[18] J.A. Rossiter and M. Abdullah, "A new paradigm for predictive functional control to enable more consistent tuning," in *2019 American Control Conference*, pp. 366-371, 2019.

[19] M. Abdullah and J.A. Rossiter, "Using laguerre functions to improve the tuning and performance of predictive functional control," *International Journal of Control*, vol. 94, no. 1, pp. 202-214, 2021.

[20] S. Idros, M. Abdullah, S.F. Toha, and M.S. Md Said, "Modelling of vehicle longitudinal dynamics for speed control," *Journal of Advance Research in Applied Mechanics*, vol. 123, no. 1, pp. 56-74, 2024.

[21] J. Richalet and D. O'Donovan, *Predictive functional control: Principles and industrial applications*. Springer Science & Business Media, 2009.

[22] R. Rajamani and R. Rajamani, "Longitudinal vehicle dynamics," in *Vehicle Dynamics and Control. Mechanical Engineering Series*. Springer, Boston, pp. 87-111, 2012.