Journal of Engineering Science and Technology
Vol. 16, No. 1 (2021) 681 - 695

ADAPTIVE CRUISE CONTROL OF A SIMSCAPE DRIVELINE VEHICLE MODEL USING PID CONTROLLER

ALI MAHMOOD*, MOHAMMED ALMAGED, MOHANAD N. NOAMAN, YAZEN HUDHAIFA SHAKIR ALNEMA

College of Electronics Engineering, Ninevah University,
AL-majmoaa Street, 41002, Mosul, Iraq
*Corresponding Author: ali.mahmood@uoninevah.edu.iq

Abstract

The arising of an autonomous vehicle has proven to reduce accidents effectively. However, such technology requires time and cost to develop, while vehicle system simulation software provides a cost-efficient, easier and safer way to analyse automotive applications. This paper presents a design methodology of Adaptive Cruise Control (ACC) system. Initially, it involves constructing a complete graphical model of the vehicle using MATLAB Simscape library. This approach will eliminate the need for mathematical derivation, model uncertainty and linearization. It will also produce a quite realistic behaviour since it takes into consideration all the key role parameters of the vehicle. Then, two PID controllers are designed to adjust the throttle and brake respectively. The designed algorithm takes into account keeping the actual distance above or equal to the safe distance by modifying the set values of the controllers according to the actual speed and the speed of the vehicle ahead. Several simulations are performed to verify controllers' capability for different driving situations. The simulation results reveal that the two PID controllers are successful in achieving the desired vehicle speed while simultaneously maintaining a safe distance. Furthermore, the results show that the system responded smoothly to the set speed in an acceptable rise and settling times with almost zero overshoot and steady state error.

Keywords: Adaptive cruise control, Autonomous vehicles, PID controller, Simscape vehicle model,.

1. Introduction

Automated vehicle control system (AVCS) is a crucial part of autonomous cars that represent the core of Intelligent Vehicle Systems (IVS) [1, 2]. Recently, the adaptive cruise control feature has been integrated into the passenger cars by using a sophisticated control and sensing elements as well as enhancing engine capabilities. The purpose of controlling vehicles autonomously is to increase drivers' safety and decrease accident rates by putting the driver and passengers out of the control loop of the driving system [3]. Simple and fully automated cruise control systems are now a part of the new architecture of modern highway systems that tend to increase freeway and traffic flow [4-6].

Recently, a cruise control system has become one of the advanced features in automobile industry. Instead of adjusting throttle and brake pedal frequently by the driver, the adaptive cruise control system can be utilized to take over the control of the car according to desired speed set by the driver. Therefore, this system has an additional benefit of reducing driver's fatigue in a long road trip [7]. Adaptive Cruise Control (ACC) is introduced as a more advanced version of a traditional cruise control system [8]. A sensory system is attached to the vehicle to detect slow moving vehicles ahead [9]. Thus, an advance adaptive cruise system allows the vehicle to adopt set speed according to both traffic environments and driver's desires.

Speed control of vehicles, that is usually referred to as the cruise control system, has been applied from the mid of the twentieth century in many companies such as Chrysler, Audi, BMW, etc. According to the 1948 discovery of a mechanical engineer community, this term is initially used in Chrysler 1958 Imperial. Based on the relation between ground speed and rotational speed of the wheels, such system physically controls the throttle position by energizing a bi-directional screw-drive electric motor [10]. This system has the task of keeping the speed of the vehicle constant according to the desired value set by the driver. However, it has a crucial drawback at which the vehicle does not make any interaction with the environment, especially with other vehicles ahead. This means when a vehicle or a person passes by the driver has to step on the brake pedal in order to reduce the speed or even stop the vehicle completely. The new generation in the world of speed control of the vehicles is the Adaptive Cruise Control (ACC) system. The ACC system differs from the ordinary cruise control system in adjusting vehicle's speed to maintain a safe and acceptable distance to the vehicles ahead. This is usually accomplished by acquiring data from a distance sensor, Lidar or Radar, mounted on the vehicle. The first commercial ACC system that assisted by a Radar was implemented by Mercedes-Benz in 1991 [11]. A similar ACC system based on Lidar detection was employed by Mitsubishi in 1992.

Many researchers have paid great attention to the controller that is being utilized to modify the speed of the vehicle relying on both driver desire and distance to the vehicle ahead. In addition, they are concentrating on the algorithm that will be used to reduce the response time or expand the range of the speed that ACC can deal with. Several control approaches have been applied in ACC systems. These control techniques might be a PID controller, Fuzzy logic, LQR or even with only a feedforward system [12, 13]. The PID based controller has been utilized widely in Adaptive Cruise Control. A PID control approach has been introduced in [14] for improving driving stability and comfort. For more enhancement in ACC systems, the design problem can be recast into an optimization problem of which solved by

one of the searching methods. Ant Lion Optimization (ALO) is one of these methods that has been presented in [15]. A better performance has obtained for Adaptive Cruise Control in terms of settling time, rise time, peak time, maximum overshot and steady state error.

Fuzzy logic controller has also been employed widely in the field of ACC system. Fuzzy PI and PD controllers have been applied successfully for automatic cruise control system [16, 17]. Also, a self-tuning fuzzy-PID controller algorithm has been developed in [18] which combines the benefits of both PID and fuzzy logic controllers. In [19], a fuzzy-logic algorithm is used as switching and tuning supervisor of a multiple-controller framework for ACC. Driving stability and comfort are one of the key concerns of the ACC system. In [20], a fuzzy logic controller is designed for ACC system taking in consideration enhancing driving experience in term of stability and comfort. other fuzzy control systems have been employed to achieve both ACC and Stop-Go control [21, 22]. A fuzzy logic control (FLC) approach has implemented in [23] with Gravitational Search Algorithm (GSA) for optimization. Moreover, an adaptive neuro-fuzzy predictive control has proposed in [24-26]. Fuzzy can human reasoning and dealing with nonlinear and complex problems. Fuzzy rules mostly rely on the experience of the designer. However, one of the major problems of employing a fuzzy controller is that there are too many parameters to cope with which generally yielding nonoptimal control in real driving conditions [27].

Model predictive control (MPC) has received significant attention for automotive control applications. ACC based on MPC has been implemented in [28, 29] for energy-optimization. In [30], a linear MPC cooperative ACC approach has been used to reduce fuel consumption. ACC algorithm with multi-objectives (safety, energy-economy, comfort, and car following) has been reported in [31, 32] based on MPC. In [33], MPC has been proposed to design a practical ACC for traffic jam. MPC has capable of real-time multi-objective optimal control and easy to achieve accurate and optimal control. However, an accurate modelling of the driving dynamics behaviour is required in order to obtain acceptable and more realistic results. The main drawback of MPC is strongly depending on the model accuracy, which most of aforementioned techniques were using a mathematical model for the vehicle, which has approximations, and linearization for the complex complicated driving system [34]. Furthermore, MPC is requiring a certain level of future trip information, which is generally not the case in real life and it has a heavy computational burden.

This paper presents implementation of a complete graphical modelling of the vehicle comprising all of its systems using MATLAB Simulink through Simscape Driveline library. This approach will eliminate the need for mathematical derivation, model uncertainty, linearization and approximations. Also, two PID controllers will be used to control the throttle and brake respectively. PID tuner tool with the benefits of modifying system structure and performance will be utilized to achieve the parameters of both controllers. These controllers will take control actions depending on the desired speed set by the driver while maintain safe distance to the vehicle ahead.

2. Vehicle Modelling

The realistic model of the vehicle could be understood by explaining the powertrain components that start from power generated by the engine to the power delivered

by the drive shafts, differentials and finally drove to the road surface. Graphical modelling offers great flexibility in describing the structure of a system using a physical element network rather than mathematical derivations. MATLAB Simulink gives a complete library for modelling and simulation of rotational and translational mechanical systems [35]. It also facilitates control systems design and testing of complex physical systems. This model simulates a complete vehicle comprising the engine, drivetrain, four-speed transmission gearbox, tires, and longitudinal vehicle dynamics using Simscape Driveline library [36]. In terms of this model, the torque generated by the engine, according to throttle level, is transferred to the torque converter, which will deliver it to a 4-speeds transmission gearbox. The differential connected to the gearbox splits the received torque equally into the two rear wheels, which are driving the whole vehicle body. Figure 1 shows the complete vehicle model, including engine, transmission and vehicle body, constructed in MATLAB Simulink.

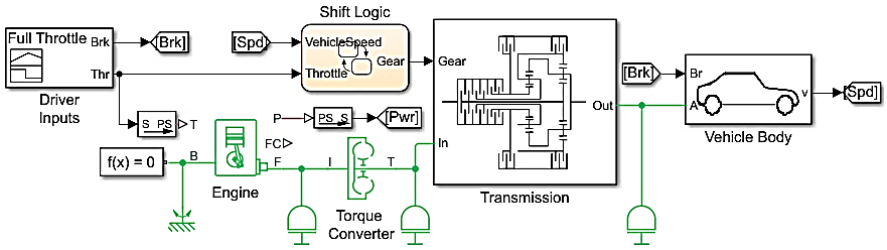


Fig. 1. Constructed vehicle model.

The engine block simulates an internal combustion engine with the throttle, time lag and rotational inertia. The physical input port T specifies the normalized engine throttle level (0 - 1) and hence states the engine torque at fixed engine speed. Model parameterization of the engine block includes specifying engine type, maximum power, the speed at maximum power, etc. It also requires specifications of the engine dynamic behaviour such as engine inertia and time constant. The other important block in the vehicle's Driveline is a torque converter. It couples two axes by the mean of hydrodynamic action of a viscous fluid to transfer torque and rotational motion. The four-speed transmission subsystem comprises five friction clutches connected to two planetary gears as shown in Fig. 2. Planetary gears are commonly used in transmission systems as they sustain high gear ratios in a compact space. The ratio of the selected gear defines the clutch schedule, shown in Fig. 3, that states which clutch must be engaged. The shift logic subsystem is a state machine architecture that controls the sequence of gears changing. This system calculates the upshift, downshift thresholds depending on the currently selected gear, and throttle input level. Then, it compares these thresholds with the current vehicle speed to decide whether the car needs to shift gears or remain engaged on the same gear.

The vehicle subsystem, as shown in Fig. 4, is mainly consisted of differential, friction brakes, tires, and vehicle body. The differential is employed to equally split the incoming torque from the transmission gearbox into left and right tire of the rear driven axle. The double - shoe brake push against the rotating drum to generate a braking action that causes the rotating drum to decelerate. The tire model provides the rolling action of the vehicle. This module takes into consideration the thrust

produced by the tire, normal force acting on the tire, tire slip, wind velocity and road inclination angle.

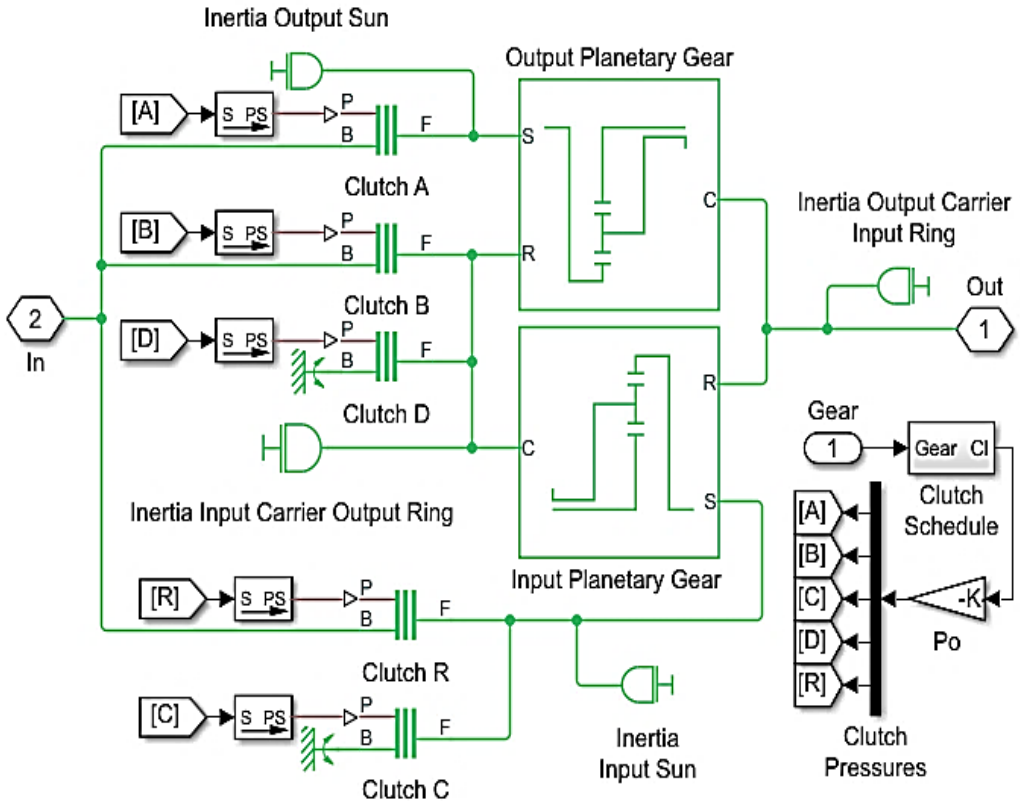
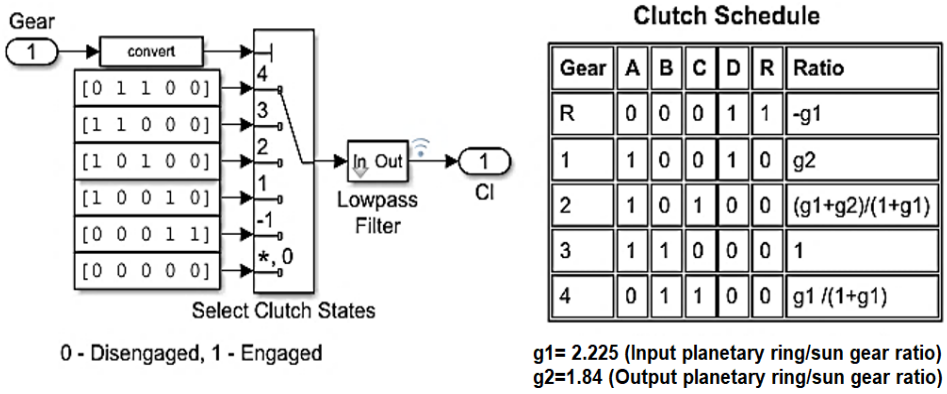


Fig. 2. Transmissions subsystem.



Clutch Schedule

Gear	A	B	C	D	R	Ratio
R	0	0	0	1	1	-g1
1	1	0	0	1	0	g2
2	1	0	1	0	0	(g1+g2)/(1+g1)
3	1	1	0	0	0	1
4	0	1	1	0	0	g1 /(1+g1)

g1= 2.225 (Input planetary ring/sun gear ratio)
g2=1.84 (Output planetary ring/sun gear ratio)

Fig. 3. Clutch schedule subsystem.

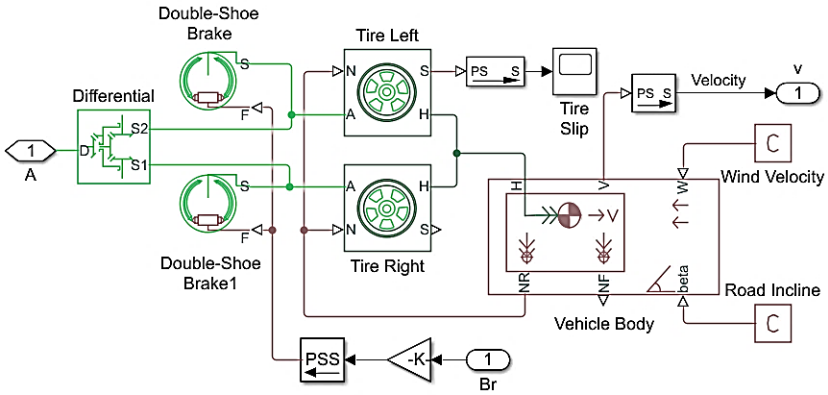


Fig 4. Vehicle body subsystem.

The default MATLAB parameters of the vehicle, listed in Table 1, are utilized to run the simulations. This comprises several specifications of the engine's performance, transmission and vehicle's body size and dimension.

Table 1. Vehicle's parameters.

Engine Parameter	Value	Body Parameter	Value
Maximum power (Watt)	1.5e+5	**Mass (kg)**	1200
Maximum speed (RPM)	6e+3	**Horizontal distance from CG to front axle (m)**	1.4
Speed at maximum power (RPM)	4.5e+3	**Horizontal distance from CG to rear axle (m)**	1.6
Initial speed (RPM)	960	**CG height above ground (m)**	0.5
Stall speed (RPM)	500	**Frontal area (m^2)**	3
Engine inertia (kg.m^2)	0.08	**Drag coefficient**	0.4
Engine time constant (sec)	0.4	**Wind velocity (m/s)**	5

3. Controller Design

The first step of designing a controller is describing the sequence of operations, shown in Fig. 5, required to control the whole process. Designing the algorithm requires two controllers, the first controller is used to adjust the throttle and the other one for controlling the brake action. This is necessary to ensure that the actual speed (*v_actual*) exactly follows the set speed (*v_set*) which has been chosen depending on desired speed (*v_desired*) and the speed of the vehicle ahead (*v_ahead*). The designed algorithm also takes the actual distance into consideration keeping a safe distance (*d_safe*) from the vehicle ahead. When the *v_ahead* is reduced suddenly, *v_actual* needs time to be reduced and as a result this will reduce the distance between the two vehicles (*d_ahead*). The speed of the vehicle ahead is calculated as follows:

$$v_{ahead} = v_{actual} + v_{relative} \quad (1)$$

$$v_{relative} = \frac{\Delta\, d\_ahead}{\Delta\, time} \quad (2)$$

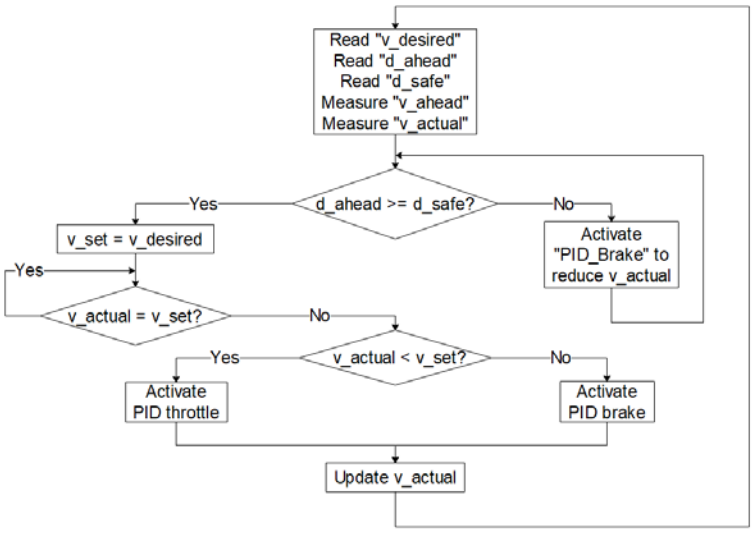


Fig. 5. Adaptive cruise control flowchart.

The structure of the proposed Proportional Integral Derivative with filter controller is shown in Fig. 6.

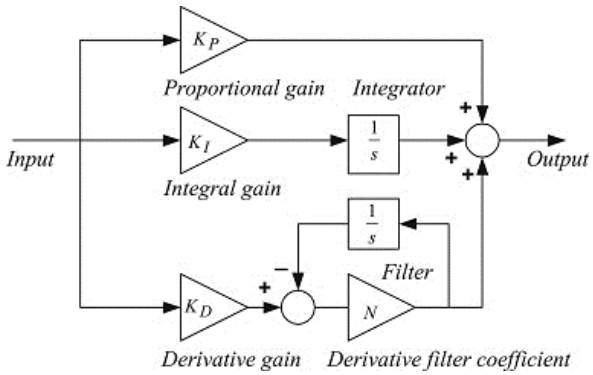


Fig. 6. Structure of PID controller with derivative filter.

The transfer function of the proposed controller can be written as follow:

$$c(s) = KP + \frac{KI}{s} + \frac{KD\,s}{\frac{s}{N}+1} \tag{3}$$

$$c(s) = \frac{(KP+KD.N)s^2+(KI+KP.N)s+KI.N}{s(s+N)} \tag{4}$$

As the graphical model of a vehicle has several nonlinear subsystems which makes the overall system is highly nonlinear. MATLAB auto-tuning failed to produce an initial value of the controller gains (*KP, KI, KD*). Therefore, the system needs to be linearized so that the PID controller can be tuned accordingly. The PID tuner tool will attempt to linearize the system and produce a new plant model of the vehicle. This can be accomplished by giving an input signal and obtaining an output response of the identified model of the vehicle. It is required to define the

specifications of this input signal. These are the sampling time, offset, onset Lag, stop time and the amplitude. Then, two signals will be generated, the identification data and identified plant with "adjuster point" which can be moved manually to further improvement. The default identified plant structure is represented as one pole system. The structure of the identified plant is changed to two underdamped pairs to produce a more realistic plant model as shown in Fig. 7. Auto values will be given for the damping ratio *(ζ)* and natural frequency (*Wn*) and these values can be auto-tuned to obtain a better response (Fig. 8). Further manual tuning could be done by adjusting the speed and robustness of the system response. By accomplishing all the previous steps, the gain values of the PID controller will be updated automatically.

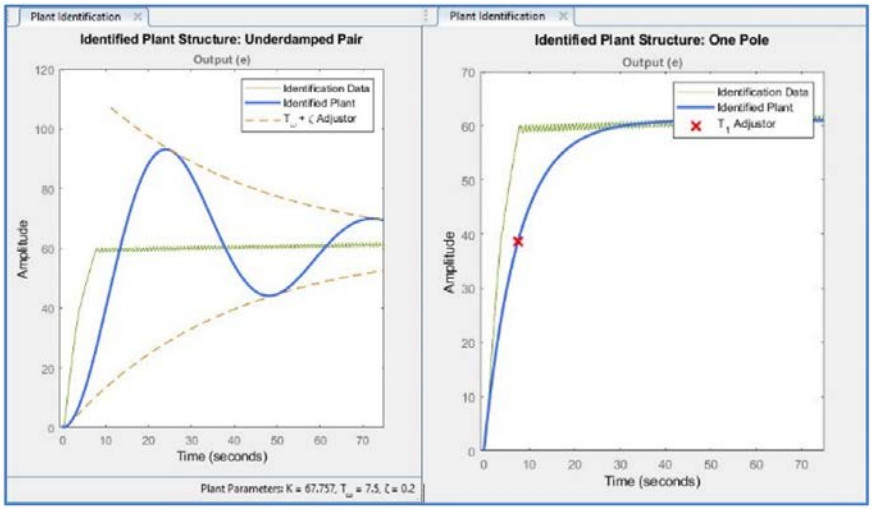


Fig. 7. Plant and identification data structure.

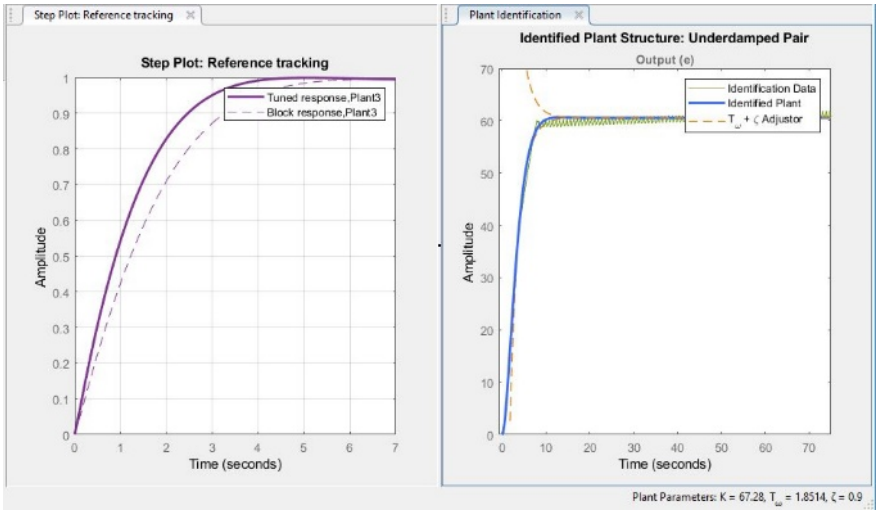


Fig. 8. Fine-tuned plant response.

The same procedure is repeated for the brake controller. It has been noted that the controller values of both brake and throttle are not the same. This is mainly due to the fact that the vehicle has different acceleration and deceleration characteristics. The parameters of the throttle and brake controllers produced by the auto tuner tool are shown in Table 2.

Table 2. PID controllers' parameters.

	Throttle PID	Brake PID
KP	0.011854	0.001159
KI	0.000968	0.000005796
KD	-0.010537	-0.02104796
N	0.15447078	0.00756188

Using PID tuner tool always guarantees that the designed controller is stable even though one of the gains might have a negative value. As shown in Table 2, derivative gain, *KD*, has a negative sign and according to Eq. (4), the system is stable as long as $KP > KD.N$. In other words, the Coefficient of s^2 is positive. The overall structure of the vehicle along with the Adaptive Cruise Control system is shown in Fig. 9. After reading the desired speed selected by the driver, several parameters are needed to be measured. For Instance, a radar sensor is required to measure the distance to the vehicle ahead and hence set the vehicle speed accordingly. The signal of the radar sensor is simulated in MATLAB, which can take different patterns to test the performance of the controllers at different driving situations. According to given input value from the radar and the desired speed chosen by the driver, the algorithm will choose the set speed of the vehicle. The second stage is to select the appropriate controller, throttle controller or brake controller, depending on the actual speed and the set speed of the vehicle.

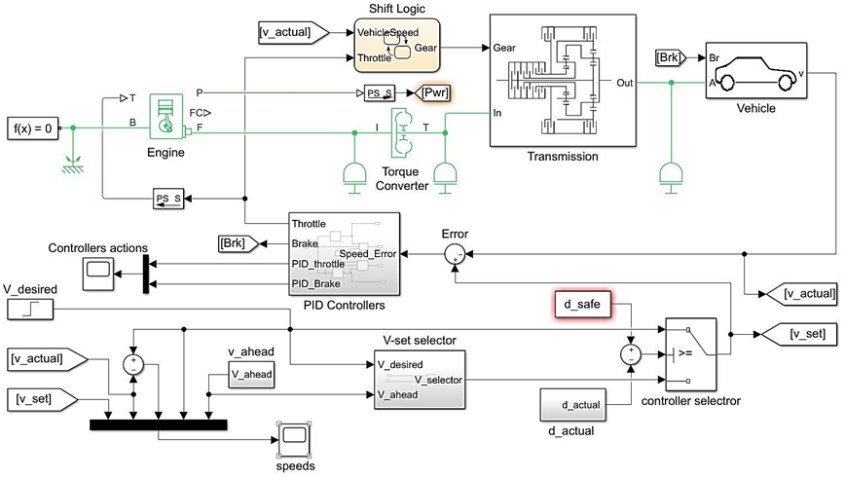


Fig. 9. ACC system structure.

4. Simulation and Results

After designing the PID controllers, several simulations have been performed using MATLAB Simulink. The solver of the simulation has been chosen automatically

with variable step size. These multiple simulations executed to verify the controllers' capability and performance for three random desired speeds (40, 60, 70) mph. The first test was implemented at *v_desired* = 40 mph. Figure 10 shows that the actual speed of the vehicle trying to follow the desired speed selected by the driver when the speed of the vehicle ahead is greater than the desired speed. Once the *v_ahead* drops down the desired value, the *v_actual* will not follow the desired speed selected by the driver, but it will follow the speed of the vehicle ahead (*v_ahead*) to keep a safe distance and avoid collision.

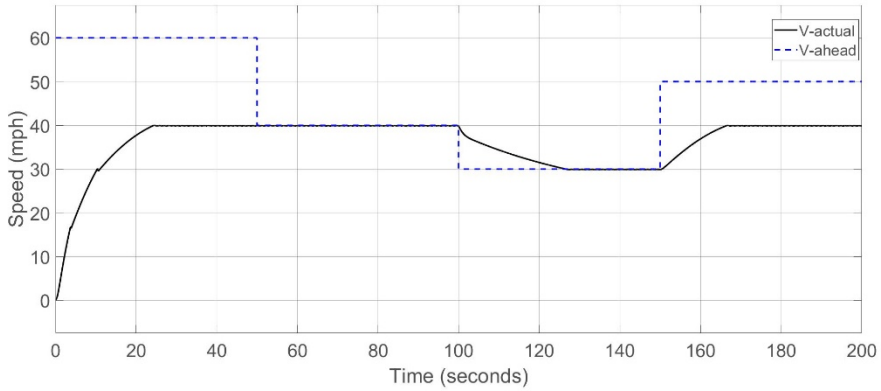


Fig. 10. Actual speed response at 40 mph desired speed.

Another test has been accomplished at a different desired speed. This test will also present the automatic change in gear state according to the actual speed of the vehicle. Figure 11 shows the response of the vehicle speed to a desired 60 mph along with that the gear selection based on the required speed and engine torque.

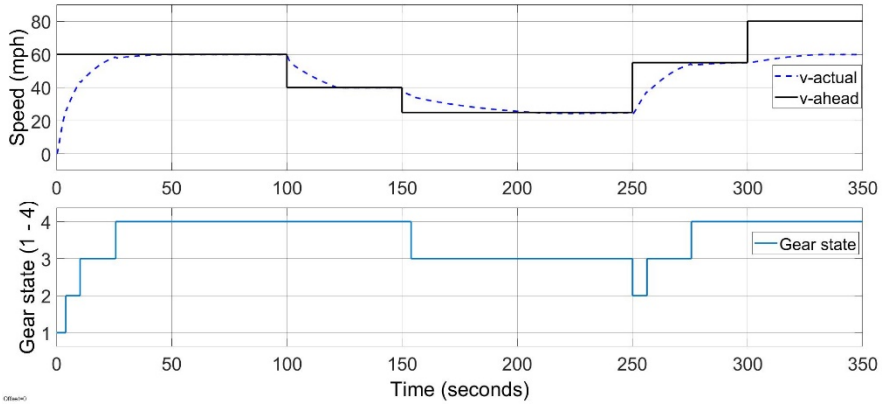


Fig. 11. Actual speed response at 60 mph desired speed with gear state.

The last test shown in Fig. 12 is performed at high desired speed while the change in the *v_ahead* is also high. This test has more difficult control situation than the rest of the tests.

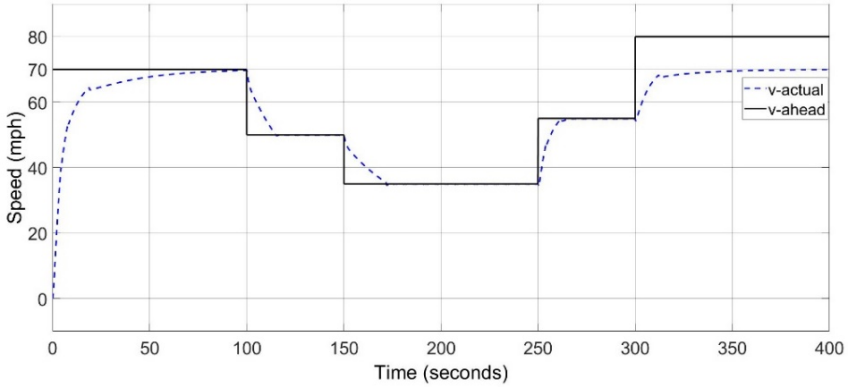


Fig. 12. Actual speed response at 70 mph desired speed.

The simulation results show that the two controllers are working in parallel to accelerate and decelerate vehicle speed while simultaneously ensuring a safe distance. Thus, the *v_set* of each controller is changing regularly according to the desired speed and the speed of the vehicle ahead. It can be noticed that the system smoothly responded to the desired speed in an acceptable rise and settling time with almost zero overshoot and steady state error that meets the design requirements. However, there are some small notches in the speed cure which are due to the change in gear state. The time response specifications of the system are shown in Table 3.

Table 3. Time response specifications.

	V_desired 40 mph	**V_desired 60 mph**	**V_desired 70 mph**
Percent Overshoot, *%OS*	0.505 %	0.861 %	0.883 %
Rise time, *Tr* (sec)	16.193	17.107	18.391
Settling time, *Ts* (sec)	22	33.5	46
Peak time, *Tp* (sec)	25	48	70
Steady state error, *Ess*	0.0005	0.001	0.08

It is important to make sure that the change in the vehicle speed happens rapidly to keep the safe distance in its acceptable range. If the safe distance is less than the designed value, the collision avoidance algorithm will be activated. At the same time, a sudden change in the speed can cause uncomforting to the driver and passengers. Also, it may cause an accident with the vehicle behind. Therefore, a balanced time response specification should be obtained to ensure both the safety and comfortability of the driver and passengers.

The designed ACC algorithm also takes into consideration keeping the actual distance to the vehicle ahead either above or equal to the safe distance. The safety distance value depends on the vehicle's actual speed. It is taken as a ratio of this speed plus an offset to avoid zero safety distance at stop condition while the actual distance is simulated using ramp signals with multiple slopes for different acceleration and deceleration scenarios. The actual speed of the vehicle, shown in Fig. 13, always follows the set speed of the controller, which is either equal to the desired speed selected by the driver or the speed of the vehicle ahead. The set speed

is equal to the desired speed when the actual distance is greater than the safety distance as well as the desired speed is higher than the speed of the vehicle ahead. While the set speed is equal to vehicle's ahead speed when the desired speed is greater than the speed of the vehicle ahead and also the actual distance is less than the safety distance. For example, during the period of the (0-35) second, the actual speed follows the speed of the vehicle ahead as the safety distance is less than the actual distance. Gradually, the gap between the two vehicles starts to get bigger until it settles down to a constant value. However, once the speed of the vehicle ahead increases suddenly, the actual speed rises too until it reaches the desired value. Similarly, as the vehicle's ahead speed drops, the set speed is returned to its initial value and the vehicle responds accordingly. The relative speed between the two vehicles shows a gradual increase or decrease in the speed of the vehicle without any sudden changes that may discomfort driving journey.

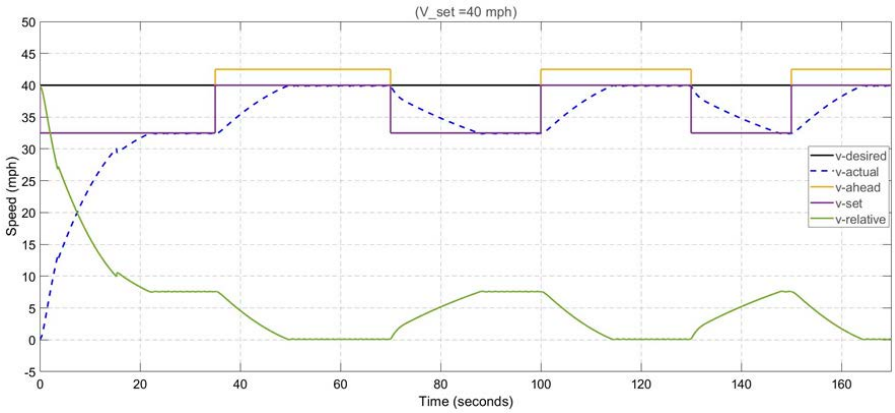


Fig. 13. ACC speed response.

5. Conclusion

In this paper, an adaptive cruise control system has been designed and simulated. Initially, a complete graphical model for a real vehicle is created via the MATLAB Simscape library. Unlike derived mathematical models of vehicles that does not take into account a mass properties or external force effects, this kinematic model produces a quite realistic behaviour as it takes into consideration several external forces such as wind velocity as well as vehicle's Powertrain characteristics. Then, two PID controllers for the throttle and brake have been designed and simulated in MATLAB Simulink.

The designed adaptive algorithm takes into an account keeping the actual distance above or equal to a safe distance by modifying the set speed of the controllers according to the actual and the vehicle ahead speeds. Several simulations were performed to evaluate the performance and effectiveness of the designed PID for ACC systems. A reasonable time response specification was obtained which ensures both the safety and comfortability of driving. However, it is important to stress out that the acceleration and deceleration of both vehicles were not taken into consideration which might affect the value of the actual distance during the actual execution of the algorithm.

Nomenclatures

d_ahead	The actual distance between the two vehicles, mile
d_safe	The safe distance between the two vehicles mile
Ess	Steady state error
KD	Derivative gain
KI	Integral gain
KP	Proportional gain
Ts	Settling time, sec
Tp	Peak time, sec
Ns	Filter coefficient in the derivative filter of the PID controller
%OS	Percent Overshoot
Tr	Rise time, sec
v_actual	Actual speed of the vehicle, mph
v_ahead	The speed of the vehicle ahead, mph
v_desired	The speed selected by the driver, mph
v_relative	The relative speed between the two vehicles, mph
v_set	The set speed that vehicle should have, mph
Wn	Natural frequency, rad/sec

Greek Symbols

ζ	Damping ratio

Abbreviations

ACC	Adaptive Cruise Control
AVCS	Automated Vehicle Control System
IVS	Intelligent Vehicle Systems
PID	Proportional, Derivative, Integral

References

1. González, D.; Pérez, J.; Milanés, V.; and Nashashibi, F. (2015). "A review of motion planning techniques for automated vehicles," *IEEE Transactions on Intelligent Transportation Systems*, 17, 1135-1145.
2. Bimbraw, K. (2015) "Autonomous cars: Past, present and future a review of the developments in the last century, the present scenario and the expected future of autonomous vehicle technology," in *2015 12th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, 191-198.
3. Alnema, Y.; Almaged, M.; and Noaman, M. (2020). Driverless model cars: A review and analysis of autonomous vehicle literature on technology and application. *International Review of Automatic Control (IREACO)*, 13 (2), 84-92.
4. Chen, N.; Wang, M.; Alkim, T.; and Arem, B.V. (2018). A robust longitudinal control strategy of platoons under model uncertainties and time delays. *Journal of Advanced Transportation*, vol. 2018.
5. Gilimyanov, R.F. (2011). Recursive method of smoothing curvature of path in path planning problems for wheeled robots. *Automation and Remote Control*, 72, 1548-1556.
6. Almaged, M.; Dawood, Y.S.; and Mahmood, A. (2018). Autonomous model vehicles: Signal conditioning and digital control design. *International Journal of Engineering and Innovative Technology (IJEIT)*, 8(3), 7.

7. Sivaji, V.V.; and Sailaja, M. (2013). Adaptive cruise control systems for vehicle modeling using stop and go manoeuvres. *International Journal of Engineering Research and Application*, 3(4), 2453-2456.
8. Payman, S.; Ordys, A.; and Askari, M.R. (2012). Adaptive cruise control with stop&go function using the state-dependent nonlinear model predictive control approach. *ISA transactions*, 51(5), 622-631.
9. Shakouri, P.; Czeczot, J.; and Ordys, A. (2012). Adaptive cruise control system using balance-based adaptive control technique. *17th International Conference on Methods & Models in Automation & Robotics (MMAR)*, 510-515.
10. Dawood, Y.S.; Mahmood, A.K.; and Ibrahim, M.A. (2018). Comparison of PID, GA and Fuzzy Logic Controllers for Cruise Control System. *International Journal of Computing and Digital Systems*, 7, 311-319.
11. Guvenc, B.A.; and Kural, E. (2006). Adaptive cruise control simulator: A low-cost, multiple-driver-in-the-loop simulator. *IEEE Control Systems Magazine*, 26, 42-55.
12. Diba, F.; Arora, A.; and Esmailzadeh, E. (2014). Optimized robust cruise control system for an electric vehicle. *Systems Science & Control Engineering: An Open Access Journal*, 2, 175-182.
13. Chaudhari, K.; Joshi, A.; Kunte, R.; and Nair, K. (2013). Design and development of roll cage for an all-terrain vehicle. *International Journal on Theoretical and Applied Research in Mechanical Engineering*, 2, 49-54.
14. Pan, Z.; Bao, H.; Pan, F.; and Xu, C. (2016). An intelligent vehicle based on an improved PID speed control algorithm for driving trend graphs. *International Journal of Simulation System*, 17, 19.1-19.7.
15. Pradhan, R.; Majhi, S.K.; Pradhan, J.K.; and Pati, B.B. (2017). Performance evaluation of PID controller for an automobile cruise control system using ant lion optimizer. *Engineering Journal*, (5), 347-361.
16. Pallab, M.; Patra, S.M.; and Mahapatra, K. (2015). Design and implementation of fuzzy approximation PI controller for automatic cruise control system." *Advances in Artificial Intelligence*, 2015, 1-7.
17. Prabhakar, G.; Selvaperumal, S.; and Pugazhenthi, P.N. (2019). Fuzzy PD plus I control-based adaptive cruise control system in simulation and real-time environment. *IETE Journal of Research*, 65, 69-79.
18. Gong, L.; Luo, L.; Wang, H.; and Liu, H. (2010). Adaptive cruise control design based on fuzzy-PID.*2010 International Conference on E-Product E-Service and E-Entertainment*. Henan, China, 1-4.
19. Abdullah, R.; Hussain, A.; Warwick, K.; and A.J. N. Zayed. (2008). Autonomous intelligent cruise control using a novel multiple-controller framework incorporating fuzzy-logic-based switching and tuning. *Neurocomputing*, 71(13-15), 2727-2741.
20. Ko, S.J.; and Lee, J.J. (2007). Fuzzy logic based adaptive cruise control with guaranteed string stability. *2007 International Conference on Control, Automation and Systems*. Seoul, South Korea, 15-20.
21. Tsai, C.C.; Hsieh, M.M.; and Chen, C.T. (2010). Fuzzy longitudinal controller design and experimentation for adaptive cruise control and stop and go. *Journal of Intelligent & Robotic Systems*, 59, 167-189.

22. Sathiyan, S.; Paul, S.; Kumar, S.; and Selvakumar, A.I. (2016). Optimized fuzzy logic-based adaptive cruise control vehicle for urban and highway driving patterns. *In Emerging Research in Computing, Information, Communication and Applications*, 319-331.
23. Bostanian, M. (2017). Fuzzy-GSA based control approach for developing adaptive cruise control. *ADMT Journal 10*, 4, 7-19.
24. Zikrija, A.; Cernica, E.; and Omanovic, S. (2019). Adaptive neuro-fuzzy inference system based modelling of vehicle guidance. *Journal of Engineering Science and Technology*, 14(4), 2116-2131.
25. Lin, Y.C.; Nguyen, H.L.T.; and Wang, C.H. (2017). Adaptive neuro-fuzzy predictive control for design of adaptive cruise control system. *In 2017 IEEE 14th International Conference on Networking, Sensing and Control (ICNSC)*. Calabria, Italy, 767-772.
26. Lin, Y.C.; and Nguyen, H.L.T (2019). Adaptive neuro-fuzzy predictor-based control for cooperative adaptive cruise control system. *IEEE Transactions on Intelligent Transportation Systems*, 21(3), 1054-1063.
27. Yinglong, H.; Ciuffo, B.; Zhou, Q.; Makridis, M.; Mattas, K.; Li, J.; Li, Z.; Yan, F.; and Xu, H. (2019). Adaptive cruise control strategies implemented on experimental vehicles: A review. *IFAC-PapersOnLine*, 52(5), 21-27.
28. Andreas, W.; Görges, D.; and Lin, X. (2017). Energy-optimal adaptive cruise control based on model predictive control. *IFAC-PapersOnLine*, 50(1), 12563-12568.
29. Andreas, W.; Görges, D.; and Lin, X. (2018). Energy-optimal adaptive cruise control combining model predictive control and dynamic programming. *Control Engineering Practice*, 72, 125-137.
30. Thomas, S.; and Re, L.D. (2013). A model predictive cooperative adaptive cruise control approach. *Proceedings of the American Control Conference*. 1374-1379.
31. Luo, L.H.; Liu.; Li, P.; and Wang, H. (2010). Model predictive control for adaptive cruise control with multi-objectives: Comfort, fuel-economy, safety and car-following. *Journal of Zhejiang University SCIENCE A*, 11(3), 191-201.
32. Zhao, R.C.; Wong, P.K.; Xie, Z.C.; and Zhao, J. (2017). Real-time weighted multi-objective model predictive controller for adaptive cruise control systems. *International Journal of Automotive Technology*, 18, 279-292.
33. Taku, T.; and Akasaka, D. (2018). Model predictive control approach to design practical adaptive cruise control for traffic jam. *International Journal of Automotive Engineering* , 9(3), 99-104.
34. Kuo, P.H.; Krishnamurthy, A.; and Malmborg, C.J. (2008). Performance modelling of autonomous vehicle storage and retrieval systems using class-based storage policies, *International Journal of Computer Applications in Technology*, 31, 238-248.
35. Alegre, S.; Míguez, J.V.; and Carpio, J. (2017). Modelling of electric and parallel-hybrid electric vehicle using MATLAB/Simulink environment and planning of charging stations through a geographic information system and genetic algorithms. *Renewable and Sustainable Energy Reviews*, 74, 1020-1027.
36. Documentation S. (2020). Simulation and Model-Based Design [Internet]. MathWorks; Available from: https://www.mathworks.com/help/physmod/sdl/ug/about-the-complete-vehicle-model.html