IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 10 August 2023, accepted 30 August 2023, date of publication 4 September 2023, date of current version 11 September 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3311850

RESEARCH ARTICLE

Design of DDPG-Based Extended Look-Ahead for Longitudinal and Lateral Control of Vehicle Platoon

ANGGERA BAYUWINDRA[1], LEON WONOHITO[1], AND BAMBANG R. TRILAKSONO[1,2], (Member, IEEE)

[1]School of Electrical Engineering and Informatics, Bandung Institute of Technology, Bandung 40132, Indonesia
[2]University Center of Excellence—Artificial Intelligence on Vision, NLP and Big Data Analytics (U-CoE AI-VLB), Bandung Institute of Technology, Bandung 40132, Indonesia

Corresponding author: Anggera Bayuwindra (bayuwindra@itb.ac.id)

This work was supported by the Bandung Institute of Technology (ITB) under Penelitian Pengabdian Masyarakat dan Inovasi (PPMI) Research Program under Grant 905y/IT1.C12/KU/2022.

ABSTRACT This paper presents a novel Deep Deterministic Policy Gradient (DDPG) algorithm with extended look-ahead approach for longitudinal and lateral control of vehicle platooning. The DDPG algorithm is adapted due to its ability to fit nonlinear system and to handle continuous control environment. Moreover, the dynamic input inversion is introduced to reduce domain of the action space from DDPG output. The existing look-ahead approach is considered as a cost-effective approach since it uses the available information from on-board sensors and is effective against the loss of lane markings. However, the approach is known to suffer from cutting-corner phenomenon. To address cutting-corners, we introduce the extended look-ahead approach and derive the true-local error states using the already available information from lidar and V2V communication. The robustness and performance of DDPG-based extended look-ahead controller is investigated by means of simulations and validated through experiments on a Donkey Car platform. The simulations and experiments with Donkey Car show that the DDPG-based extended look-ahead algorithm can provide an efficient control strategy for longitudinal and lateral maneuvers without the requirement of path information.

INDEX TERMS Deep deterministic policy gradient (DDPG), reinforcement learning control, longitudinal and lateral control, vehicle platooning, vehicle following.

I. INTRODUCTION

The rapid development of automotive industries and the needs of mobility nowadays brings benefits and also simultaneously brings challenges and problems to the transportation. The increase of vehicles that is not balanced with the increase of road capacity has led to traffic accidents, congestion, and environmental problem. To cope with this problem, one of the solution proposed is to increase the capacity of existing roads and infrastructure. However, the development of roads and infrastructure is considered costly, arduous, and time-inefficient. Thus, an alternative solution of increasing the traffic flow is proposed. One way to increase the traffic flow is by reducing the distance between vehicles by means of vehicle platooning, which can be defined as grouping several vehicles that formed a compact formation and drive at a small inter-vehicle distance. An autonomous vehicle is then developed to enables a vehicle to drive at a closer inter-vehicle distance than a human driver could. Adaptive Cruise Control (ACC), as a part of Advanced Driver Assistance System (ADAS), is first invented as a comfort system that enables a vehicle to automatically regulate its speed to maintain a safe distance from its preceding vehicle. ACC system in a vehicle utilizes radar, laser, or a camera to detect the vehicle

The associate editor coordinating the review of this manuscript and approving it for publication was Junho Hong.

ahead, allowing it to brake when the inter-vehicle distance is too near, or accelerate when the inter-vehicle distance is too far. ACC technology is regarded as one of the key component of autonomous or intelligent vehicles. Along its development, the functionality of ACC was then extended to Cooperative Adaptive Cruise Control (CACC), that allows a vehicle to receive information from other neighboring vehicles using a Vehicle-to-Vehicle (V2V) communication. CACC realizes automated longitudinal vehicle control by adding the preceding vehicle's velocity, acceleration, or steering angle that are used in a feed-forward loop [1]. The invention of ACC/CACC has led to the realization of automated longitudinal control. However, to achieve a level 2 autonomous vehicle, in which both steering and acceleration/deceleration are automated, the lateral control of vehicle has to be taken into account.

One proposed approach to realize both longitudinal and lateral control of autonomous vehicle is to separate the control problem into two independent subsystems [2]. In this approach, the ACC/CACC system regulates the longitudinal control [3], while a lane keeping assist system regulates the lateral control [4]. Typically, a lane keeping controller is developed based on the path following method, where a vehicle detect a reference path (e.g., lane markings or magnetic markings embedded in roads) using magnetic sensors or cameras. The control objective is then to calculate a steering input that minimizes the distance from the position of vehicle to a reference path. In this lane keeping system, the clarity of lane markings is important for the accuracy of path following. The drawback of this path following method is when the lane markings are of bad quality, or even not available, resulting in the difficulty of tracking the correct lane which then may lead to accidents. From the platooning perspective, when a small inter-vehicle distance is desired, an accurate measurement of lane markings is not always possible due to the obstructed view from the preceding vehicle. Moreover, by treating the longitudinal and lateral motion independently, the application of this decomposed approach is limited to situations with small steering angles or low speeds [5].

To achieve a combined control for longitudinal and lateral motion that does not depend on lane markings, a vehicle following method is adapted. In this method, a look-ahead approach is usually used by a vehicle to follow its preceding vehicle [6]. By utilizing the already available information from the ACC/CACC setup, this approach is considered as a cost-effective and feasible solution. However, the implementation of the look-ahead approach results in cutting-corner phenomenon for the lateral movement of vehicles. Authors in [7] and [8] proposed an extended look-ahead method that compensates the cutting-corners, where the method is then applied on the vehicle platoon with kinematic vehicle model. However, the kinematic vehicle model is considered inaccurate for a dynamic environment in longitudinal and lateral maneuvers. Moreover, these two papers considered the needs of global and semi-local positions of vehicles, which in some situation cannot be accurately measured. In [9], the trajectory estimation of the preceding vehicle is designed to compensate the corner-cutting problem. Using a dynamic vehicle model, a model predictive control is applied to fulfill the longitudinal and lateral control. The extended look-ahead approach is also adapted in [10], where the nonlinear dynamics of the vehicle is taken into account. However, the physical parameters of vehicles are difficult to be measured and estimated, and the performance of the proposed controller in these papers for a heterogeneous vehicle platoon was not yet investigated.

The rapid development of artificial intelligence technology, especially Reinforcement Learning (RL), has brought benefits to autonomous driving [5], [11], [12]. The application of artificial neural networks for tracking and autonomous vehicles can be traced back to 1990s, see, e.g., [13] and [14]. In these papers, data are provided by ultrasonic sensors and used as an input to a feedforward network. Another hybrid approach, which combines RL and fuzzy logic, for longitudinal vehicle control is proposed in [15]. In [16], the authors propose an actor-critic algorithm, which use a model-free value- and policy-based RL algorithm to solve longitudinal control problem. Q-learning algorithm, as a subfield of RL, is a model-free algorithm that learn value of an action in a particular state. In [17], a deep Q-learning algorithm, which uses experience replay from a random sample of prior action instead of the most recent action, is proposed to control braking as a part of collision avoidance system. The authors in [18] propose a combined approach of supervised learning and deep Q-learning algorithm. The lateral and longitudinal control are then treated as a dependent system and the objective is formulated as a velocity control and lane keeping control. In [19], the authors propose a model-based policy iteration algorithm to solve the Hamilton-Jacobi-Bellman equation for a linearized dynamic vehicle model, and adaptive dynamic programming is implemented for the data-driven policy iteration. It is also important to note that the automated driving can be classified as a control problem for a continuous system, and the drawback of these proposed algorithms is that they can only be implemented to discrete environments. The process of discretization would lead to the inability to overcome dynamic environment in the vehicle platooning system [20].

Deep Deterministic Policy Gradient (DDPG) algorithm, which is based on a direct policy search, is then proposed to handle continuous control environment by directly output continuous action [25]. In [5], the authors propose an improved DDPG algorithm based on the double critic networks to overcome large cumulative errors. The approach is applied to a lane following method in autonomous vehicle. A DDPG-based proportional integral derivative (PID) longitudinal controller is proposed in [23] to improve the performance regarding speed tracking error. However, the proposed algorithms have not yet incorporated a vehicle dynamic model, and the longitudinal and lateral motion are still treated independently.

TABLE 1. Literature of vehicle platooning control strategy.

Reference	Year	Proposed Method	Benefits	Boundaries
[21]	2023	Observer-based secure control for vehicular platooning	Resilient under Denial-of-Service (DoS) attack	Longitudinal control only, kinematic model
[22]	2022	Sharing Deep Reinforcement Learning for vehicle platooning control	Heterogeneous platoon, multi-vehicle network	Separate longitudinal and lateral control
[11]	2022	Deep Reinforcement Learning	Heterogeneous platoon	Separate longitudinal and lateral control, kinematic model
[23]	2021	DDPG-based PID control	PID control with automated tuning, heterogeneous platoon	Separate longitudinal and lateral control, kinematic model
[5]	2021	Lane following method based on DDPG algorithm	Heterogeneous platoon	Lateral control only, kinematic model
[19]	2021	Combined longitudinal and lateral control based on Reinforcement Learning	Longitudinal and lateral control, heterogeneous platoon	Linearized dynamic model, simulation only considers one scenario
[24]	2021	Hybrid Deep Reinforcement Learning for vehicle platoon	Fast convergence, heterogeneous platoon	Longitudinal control only
[10]	2020	Coordinated longitudinal and lateral of nonlinear dynamics vehicles	Nonlinear dynamics vehicle model	Homogeneous platoon
[6]	2011	H_∞ controller	String stability	Homogeneous platoon, longitudinal control only

Several methods to control a vehicle platoon have been proposed in literature, see Table 1. Most of the studies focused on using Reinforcement Learning for the control design of vehicle platooning. However, most studies assume that the path information for the follower vehicle to follow is available, which is not true in some cases [26]. The real-time control design of vehicle platooning also needs to consider the coupled nonlinear longitudinal and lateral dynamics of vehicle. Therefore, the main contribution of this paper is the design of DDPG-based algorithm using extended look-ahead approach for a dynamic vehicle model, which can be divided into several sub-contributions. First, we introduce the control problem formulation in vehicle platooning and a nonlinear dynamic single-track vehicle model. To reduce the domain of the action space from DDPG output, we propose a dynamic input inversion method that transforms the output acceleration from DDPG algorithm to the longitudinal force input of the dynamic single-track model. With this domain reduction, the computational load of the algorithm can be reduced. To overcome cutting-corner in the look-ahead approach, we adapted the extended look-ahead approach and derive true-local error states. By this approach, the error states can be calculated using the already available information from radar and V2V communication and a global positioning information is not required. In comparison to the existing results of other RL-based controllers for vehicle following, our DDPG-based extended look-ahead controller takes the coupled dynamics of longitudinal and lateral motion into account and handles the cutting-corners without the need of path information.

The next part of this paper is formulated as follows. Section II describes the problem formulation and system model. In this section, the nonlinear dynamic single-track vehicle model, the dynamic input inversion are introduced, and the adapted extended look-ahead error is derived. In Section III, we propose the design of DDPG-based extended look-ahead for longitudinal and lateral control.

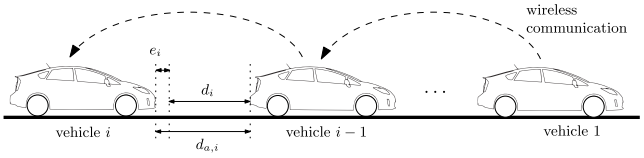

FIGURE 1. Predecessor-following platoon topology.

Section IV presents the training of the proposed algorithm in MATLAB, training results, and simulations. In Section V, we presents the implementation of the designed algorithm in a Donkey Car platform for further validation. Finally, the last section presents the summary of conclusions and future works.

II. PROBLEM FORMULATION AND SYSTEM MODEL

A. FORMULATION OF VEHICLE PLATOONING CONTROL PROBLEM

Consider a platoon of of $m \in \mathbb{N}$ vehicles with a predecessor-following topology, as depicted in Fig. 1, where $S_m = \{i \in \mathbb{N} | 1 \leq i \leq m\}$ denotes the set of all vehicles in the platoon. The first vehicle in the platoon (with index $i = 1$) is assumed to be controlled by a human driver, i.e., the vehicle can be directly velocity and steering controlled. The velocity of vehicle i is denoted by v_i, and the actual distance between vehicle $i - 1$ and vehicle i is denoted by $d_{a,i}$. With a predecessor-following topology, the main objective of vehicle i is to follow vehicle $i - 1$ at a desired distance, d_i.

The desired distance between vehicle, also known as the spacing policy, can be designed as a constant, a constant time-gap, or a variable time-gap spacing policy [27]. In this paper, the desired distance d_i is formulated as a constant time-gap spacing policy as follows

$$d_i = r_i + hv_i, \tag{1}$$

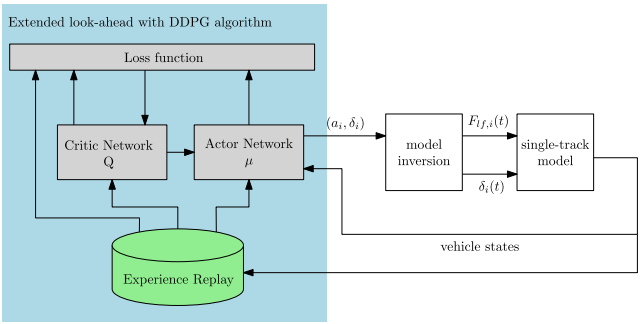

FIGURE 2. DDPG-based control structure.

where $r_i > 0$ is the standstill distance, h is the time-gap, and v_i is the velocity of vehicle i. Using this spacing policy, the inter-vehicle distance is increasing if the velocity of the follower vehicle is increasing.

Based on a predecessor-following control structure, an ego vehicle can can receive the state information (such as position, velocity, acceleration, yaw rate, and steering angle) through on-board sensors and communicate with its neighbor vehicles through a vehicle-to-vehicle (V2V) communication. The relative distance to the preceding vehicle is measured by radar, lidar, or cameras.

For a higher degree of automation in vehicle platooning, an integration of longitudinal and lateral control have to be considered. Based on the vehicle platooning control framework in Fig. 2, the DDPG controller uses the information obtained from on-board sensors and V2V communication to calculate the desired acceleration and steering angle. For a vehicle with a large acceleration and varying yaw rate, the kinematic model of vehicle is no longer valid since the lateral forces generated by the tires affect the longitudinal and lateral motion [2]. Therefore, in this paper we consider a dynamic vehicle model, and introduce the transformation of the input of a dynamic vehicle model from the output of DDPG algorithm. The transformation is implemented using a dynamic input inversion, which was first investigated in [28], and later extended to a certain class of nonlinear systems in [29] and [30]. In order to arrive at a suitable dynamic input inversion, the vehicle dynamic modeling is defined first. To fulfill the longitudinal and lateral control objective of vehicle platooning, the desired distance (1) needs to be adapted to a two-dimensional distance. In the next subsection, we design the dynamic input inversion, which then used for the formulation of the error derivation based on the two-dimensional constant time-gap spacing policy.

B. DYNAMIC MODELING OF VEHICLE

We consider the single-track vehicle model, which is commonly used to model the lateral and longitudinal dynamics of a vehicle under a normal driving conditions [2], [31]. The dynamic single-track model with a front-wheel drive is given by

$$\dot{v}_{x,i} = \frac{1}{m_i}\left(F_{lf,i}\cos\delta_i - F_{cf,i}\sin\delta_i\right) + v_{y,i}\dot{\psi}_i \tag{2}$$

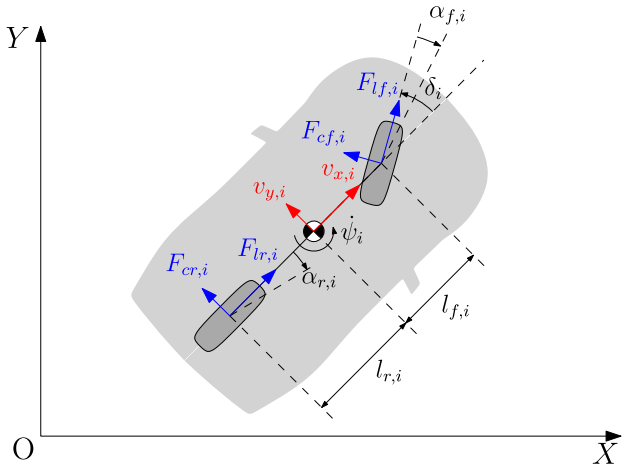

FIGURE 3. Dynamic single-track vehicle model.

$$\dot{v}_{y,i} = \frac{1}{m_i}\left(F_{lf,i}\sin\delta_i + F_{cf,i}\cos\delta_i + F_{cr,i}\right) - v_{x,i}\dot{\psi}_i \tag{3}$$

$$\ddot{\psi}_i = \frac{1}{I_i}\left(l_{f,i}F_{lf,i}\sin\delta_i + l_{f,i}F_{cf,i}\cos\delta_i - l_{r,i}F_{cr,i}\right), \tag{4}$$

where v_x, v_y, and $\dot{\psi}$ denote the longitudinal, lateral, and yaw velocity, respectively. Furthermore, m_i denotes the mass of the vehicle, I_i denotes the moment of inertia, and $l_{j,i}$, with $j \in \{f, r\}$, denotes the distance between the axle of front and rear tires, respectively, and the center of gravity of the vehicle. The front-wheel traction force of the engine is denoted by $F_{lf,i}$ and the steering angle is denoted by δ_i, which act as the inputs to the system. The cornering forces that generated by the front and rear tire, are denoted by $F_{cf,i}$ and $F_{cr,i}$, respectively, and depend on the lateral tire slip angle $\alpha_{f,i}$ and $\alpha_{r,i}$ of the respective tire, see Fig. 3.

The mapping from the lateral tire slip angle $\alpha_{j,i}$ to its respective cornering force $F_{cj,i}$, with $j \in \{f, r\}$, depends on the tire model, road condition, driving condition, and can vary in complexity. In vehicle platooning, we consider a regular driving condition with a relatively low longitudinal acceleration, steering angle rate, and small slip angle. Therefore, a linear tire model in which the cornering forces depend proportionally on their respective slip angles can be used [32], and are defined as

$$\begin{aligned} F_{cf,i} &= C_{\alpha f,i}\alpha_{f,i} \\ &= C_{\alpha f,i}\left(\delta_i - \arctan\left(\frac{v_{y,i}+l_{f,i}\dot{\psi}_i}{v_{x,i}}\right)\right) \end{aligned} \tag{5}$$

$$\begin{aligned} F_{cr,i} &= C_{\alpha r,i}\alpha_{r,i} \\ &= C_{\alpha f,i}\left(-\arctan\left(\frac{v_{y,i}-l_{r,i}\dot{\psi}_i}{v_{x,i}}\right)\right), \end{aligned} \tag{6}$$

where $C_{\alpha j,i}$, $j \in \{f, r\}$, denote the cornering stiffness coefficient of the respective tire.

C. DYNAMIC INPUT INVERSION

Let (X_i, Y_i) denote the Cartesian coordinates of the center of gravity (CoG) of vehicle i, as depicted in Fig. 4. The chassis kinematic model at the CoG in a Cartesian coordinate frame

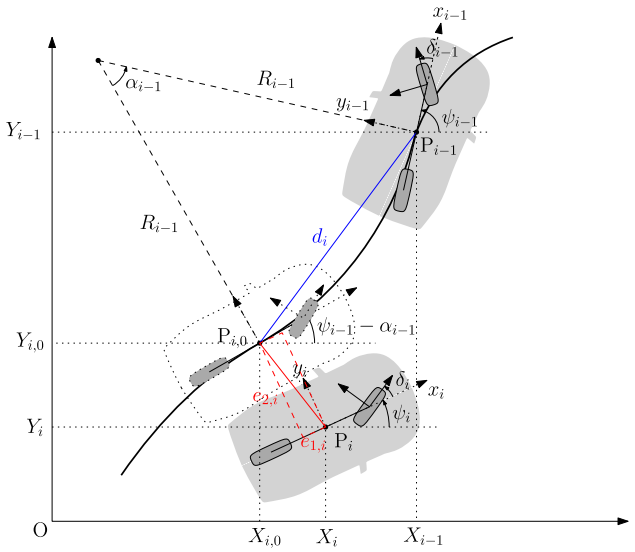

FIGURE 4. Trajectory tracking problem with extended look-ahead.

is given by

$$\dot{X}_i = v_{x,i}\cos\psi_i - v_{y,i}\sin\psi_i \tag{7}$$
$$\dot{Y}_i = v_{x,i}\sin\psi_i + v_{y,i}\cos\psi_i, \tag{8}$$

where $(v_{x,i}, v_{y,i}, \psi_i)$ dynamics are as described in (2), (3), and (4). Using (7) and (8), the velocity of the CoG, v_i, can be defined as

$$v_i = \sqrt{\dot{X}_i + \dot{Y}_i} = \sqrt{v_{x,i}^2 + v_{y,i}^2}. \tag{9}$$

The longitudinal acceleration of the CoG, a_i, is obtained by differentiating (9) with respect to time as follows

$$\dot{v}_i = \frac{v_{x,i}\dot{v}_{x,i} + v_{y,i}\dot{v}_{y,i}}{\sqrt{v_{x,i}^2 + v_{y,i}^2}} =: a_i. \tag{10}$$

By substituting (2) and (3) into (10), the longitudinal force input $F_{lf,i}$ is eventually obtained as

$$F_{lf,i} = \frac{1}{\xi_i}\left(m_i v_i a_i - F_{cf,i}\zeta_i - v_{y,i}F_{cr,i}\right) \tag{11}$$
$$\zeta_i := -v_{x,i}\sin\delta_i + v_{y,i}\cos\delta_i$$
$$\xi_i := v_{x,i}\cos\delta_i + v_{y,i}\sin\delta_i,$$

where (a_i, δ_i) are the new inputs of vehicle i with $|\delta_i| \leq \pi/4$, $F_{cj,i}$, $j \in \{f, r\}$, is as described in (5), (6), and $v_{x,i} > 0$ such that $\xi_i \neq 0$. Therefore, any control outputs of the form longitudinal acceleration and steering angle can be applied directly into the single-track model through the dynamic inversion (11). Using the new states (7), (8), and the transformed input (11), the error derivation is discussed in the next section.

D. TRUE-LOCAL EXTENDED LOOK-AHEAD ERROR DERIVATION

Using the input inversion defined in the previous section, the control to the dynamic model of vehicle i is transformed from $(F_{lf,i}, \delta_i)$ into (a_i, δ_i). To overcome the cutting-corner problem, we adapt the extended look-ahead approach in [8]. Let P_i denotes the posture of the vehicle i, $\mathrm{P}_{i,0}$ denotes the desired posture of where the vehicle i should be, and $\kappa_{i-1} = \dot{\psi}_{i-1}/v_{i-1}$ denotes the instantaneous curvature of vehicle $i-1$, see Fig. 4. The desired posture $\mathrm{P}_{i,0}$ is defined as a function of the preceding vehicle $i-1$ posture, P_{i-1}, and the angle of the circular arc formed by P_{i-1} and $\mathrm{P}_{i,0}$. The angle, denoted by α_{i-1}, is defined as

$$\alpha_{i-1} = 2\arcsin\left(\tfrac{1}{2}d_i\kappa_{i-1}\right), \tag{12}$$

where d_i is the desired inter-vehicle distance as defined in (1). By using trigonometric identities in Fig. 4, we also have

$$\sin\tfrac{\alpha_{i-1}}{2} = \tfrac{1}{2}d_i\kappa_{i-1}, \qquad \cos\tfrac{\alpha_{i-1}}{2} = \frac{\sqrt{4-d_i^2\kappa_{i-1}^2}}{2}. \tag{13}$$

From Fig. 4, it can be observed that the length of $\mathrm{P}_{i,0}\mathrm{P}_{i-1}$ is equal to the desired spacing distance d_i. Thus, the posture $\mathrm{P}_{i,0}$ in a global Cartesian coordinate is defined as

$$X_{i,0} = X_{i-1} - d_i\cos\left(\psi_{i-1} - \tfrac{\alpha_{i-1}}{2}\right) \tag{14}$$
$$Y_{i,0} = Y_{i-1} - d_i\sin\left(\psi_{i-1} - \tfrac{\alpha_{i-1}}{2}\right). \tag{15}$$

With this desired posture derivation, the control objective can be defined as bringing the posture P_i to the desired posture $\mathrm{P}_{i,0}$. We define the error state components as

$$\begin{bmatrix} e_{1,i} \\ e_{2,i} \end{bmatrix} = \begin{bmatrix} \cos\psi_i & \sin\psi_i \\ -\sin\psi_i & \cos\psi_i \end{bmatrix}\begin{bmatrix} X_{i,0} - X_i \\ Y_{i,0} - Y_i \end{bmatrix}. \tag{16}$$

By substituting (14) and (15) into (16), and by using half and double angle formulae, we eventually obtain the error state components as

$$\begin{bmatrix} e_{1,i} \\ e_{2,i} \end{bmatrix} = \begin{bmatrix} \cos\psi_i & \sin\psi_i \\ -\sin\psi_i & \cos\psi_i \end{bmatrix}\begin{bmatrix} X_{i-1} - X_i \\ Y_{i-1} - Y_i \end{bmatrix} - d_i\begin{bmatrix} \cos e_{\psi,i} & -\sin e_{\psi,i} \\ \sin e_{\psi,i} & \cos e_{\psi,i} \end{bmatrix}\begin{bmatrix} \cos\frac{\alpha_{i-1}}{2} \\ \sin\frac{\alpha_{i-1}}{2} \end{bmatrix} \tag{17}$$
$$e_{\psi,i} := \psi_{i-1} - \psi_i - \alpha_{i-1}, \tag{18}$$

where $\cos(\alpha_{i-1}/2)$ and $\sin(\alpha_{i-1}/2)$ are as defined in (13), and $e_{\psi,i}$ is the orientation error of vehicle i. It can be observed directly that the first term of the right-hand side of (17) denotes the relative longitudinal and lateral error, respectively, that can be obtained from lidar. The relative orientation $\psi_{i-1} - \psi_i$, and the yaw rate of vehicle $i-1$, $\dot{\psi}_{i-1}$, can be measured by an IMU (Inertial Measurement Unit) and lidar, while v_{i-1} is measured using on-board sensors and communicated through V2V. Using these error definitions, in the next section we define the design of DDPG-based extended look-ahead controller.

III. DESIGN OF DDPG-BASED EXTENDED LOOK-AHEAD LONGITUDINAL AND LATERAL CONTROLLER FOR VEHICLE PLATOONING

As a subfield of Reinforcement Learning (RL), Deep Deterministic Policy Gradient (DDPG) is an algorithm that uses an actor critic reinforcement to search for an optimal

policy that maximizes the expected cumulative long-term rewards. To understand how a DDPG algorithm can be applied in a standard control problem, we can view the policy inside of DDPG as a controller. Hence, the action produced by the policy acts as the control input to the system, while the states of the environment in DDPG act as the observed output of the system. In standard optimal control problems, the reward function designed in DDPG can be viewed as the objective function that the policy, or the controller, is trying to maximize. Therefore, DDPG can be seen as an algorithm that finds the solution for optimization problem of the reward function, and produces the optimal policy as an optimal control law for the system [33].

The algorithm of DDPG itself, which is an off-policy algorithm, inherits an actor-critic framework [34]. The actor is responsible for a policy, which receives the states of the environment as the input and generates an action. The critic, on the other hand, estimates the action value function, which is utilized to assess how good the actor is. The algorithm uses two deep neural networks for each actor and critic, which is powerful enough to represent control problem for a highly non-linear dynamical system [35].

To ensure the convergence of the DDPG algorithm, choosing an appropriate state space and a reward function are critical. The states of the environment should be related to the motion states of the vehicles, while the reward function should be designed to minimize the longitudinal and lateral error. For the longitudinal and lateral control problem of vehicle platooning, the observation state at time step t, s_t, are designed as a state with nine elements as follows

$$s_t = \left\{ \int e_{v,i}, e_{v,i}, v_i, \int e_{1,i}, e_{1,i}, \int e_{2,i}, e_{2,i}, \dot{e}_{1,i}, \dot{e}_{2,i} \right\}, \quad (19)$$

where $e_{v,i} = v_{i-1} - v_i$ is the velocity error, $e_{1,i}$ and $e_{2,i}$ denotes the longitudinal and lateral error, as defined in (17).

The actor network of DDPG then approximates a behavior policy, μ, with respect to the observation state (19). The output of the actor network is given by [36]

$$a_t = \mu\left(s_t|\theta^{\mu}\right), \quad (20)$$

where θ^{μ} denotes the parameter of the strategy network of making the determined action. On the other hand, the critic network approximates a value function as

$$\begin{aligned} Q &= Q\left(s_t, \tilde{a}_t|\theta^{Q}\right) \\ \tilde{a}_t &:= a_t + \mathcal{N}_t, \end{aligned} \quad (21)$$

where θ^{Q} denotes the parameter of the strategy critic network, and $\mathcal{N}_t$ represents the exploration noise.

The objective of the DDPG algorithm is to find the optimal strategy such that the cumulative reward is maximized, see Algorithm 1 [25]. The design of the reward function is critical for the DDPG-based extended look-ahead algorithm and needs to take several aspects into account: firstly, the velocity of the follower vehicle v_i must be positive to satisfy the condition of $v_{x,i} > 0$, which then implies that $\xi_i \neq 0$ as a necessary condition for (11); secondly, there should to be no collision between vehicles; and thirdly, the follower vehicle needs to be responsive to follow the maneuvers of its preceding vehicle. Therefore, as one of the main contributions in this paper, we propose the reward function R_t at every time step t, which is designed as

$$R_t = -\left(R_e + R_u\right) + R_t + R_1 + R_2, \quad (22)$$

where

$$R_e = w_e\left(50e_{1,i}^2(t) + 50e_{2,i}^2(t)\right)$$

$$R_u = w_u\left(5\dot{a}_i^2(t-1) + 5\dot{\delta}_i^2(t-1)\right)$$

$$R_t = \begin{cases} w_t, & \left|e_{2,i}\right| > 10 \text{ or } v_i < 0.5 \text{ or } x_{i-1} - x_i - d_i < 0 \\ 0, & \text{otherwise} \end{cases}$$

$$R_1 = \begin{cases} w_1, & e_{1,i}^2 < 0.01 \\ 0, & \text{otherwise} \end{cases}$$

$$R_2 = \begin{cases} w_2, & e_{2,i}^2 < 0.01 \\ 0, & \text{otherwise}, \end{cases}$$

Algorithm 1 DDPG algorithm [25]

Initiate random actor network $\mu(s|\theta^{\mu})$ with weight θ^{μ} and critic network $Q(s, a|\theta^{Q})$ with weight θ^{Q}
Initiate target network μ' with weight $\theta^{\mu'} \leftarrow \theta^{\mu}$ and Q' with weight $\theta^{Q'} \leftarrow \theta^{Q}$
Initiate replay buffer
for $i = 1$ to G **do**
Initiate random process $\mathcal{N}$ for action exploration
Obtain initial observation state s_1
for $t = 1$ to T **do**
Evaluate action $a_t = \mu(s_t|\theta^{\mu}) + \mathcal{N}_t$ with respect to the current policy and exploration noise
Implement action a_t, observe reward R_t and new state s_{t+1}
Save transition state (s_t, a_t, R_t, s_{t+1}) in replay buffer
Sample a random mini batch of N transition states (s_i, a_i, R_i, s_{i+1}) from replay buffer
Set $y_i = R_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$
Update the critic network by minimizing the loss function

$$L = \frac{1}{N}\sum_i (y_i - Q(s_i, a_i|\theta^{Q}))^2$$

Update the actor policy using sampled policy gradient:

$$\nabla_{\theta^{\mu}} J \approx \frac{1}{N}\sum_i \nabla_a Q\left(s, a|\theta^{Q}\right)|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^{\mu}} \mu\left(s|\theta^{\mu}\right)|_{s_i}$$

Update the target networks:

$$\begin{aligned} \theta^{Q'} &\leftarrow \tau\theta^{Q} + (1-\tau)\,\theta^{Q'} \\ \theta^{\mu'} &\leftarrow \tau\theta^{\mu} + (1-\tau)\,\theta^{\mu'} \end{aligned}$$

end for
end for

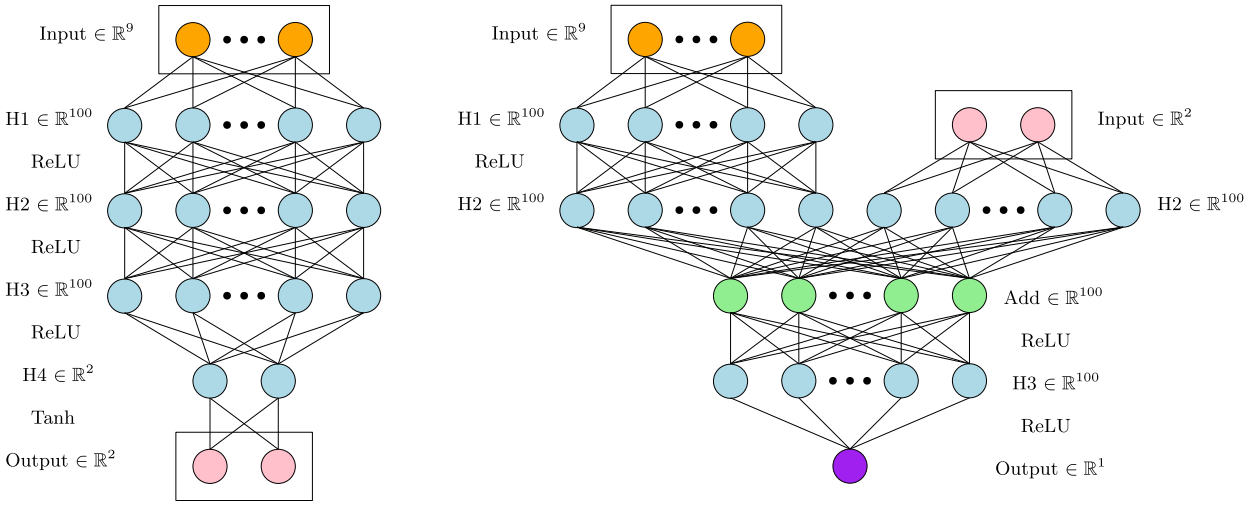

FIGURE 5. The design of DDPG neural network architecture: actor network (left) and critic network (right).

and $\dot{a}_i(t-1)$ and $\dot{\delta}_i(t-1)$ are the derivative of acceleration and steering input of the previous time step $t-1$. Negative term of reward function R_e eliminates errors to achieve the longitudinal and lateral control objective. The negative reward function R_u is designed to minimize the chattering in control effort. The reward function R_t is a logical value that terminates the training if the lateral error is too high, if the velocity of the follower vehicle is approaching zero, or if the relative inter-vehicle distance is negative. The reward functions R_1 and R_2 are logical values that encourage the agent to make longitudinal and lateral error, respectively, small. The weighting coefficients w_j, with $j = \{e, u, t, 1, 2\}$, are appropriately tuned in order to balance the importance of the components [37]. To fulfill the important aspects of vehicle following objectives, it is critical to avoid the condition of the non-zero or negative velocity and to minimize the lateral and longitudinal position errors. Hence, the weight w_t, w_1, and w_2 are chosen as 10. On the other hand, we want to avoid the aggressive control action of accelerating/decelerating, while also maintaining a smooth chattering in control effort. Hence, we set the weight w_e and w_u equals to 0.001.

IV. TRAINING OF DDPG-BASED EXTENDED LOOK-AHEAD CONTROLLER

A. TRAINING AND SIMULATION ENVIRONMENT

For the longitudinal and lateral vehicle platooning control system as described in the previous section, we utilized the actor and critic networks within the DDPG algorithm. The overall network architecture is shown in Fig. 5. The actor network consists of six layers: one input layer, one output scaling layer, and four fully connected hidden layers. Each connected hidden layer has 100 neurons with the activation function of rectified linear unit (ReLU) to accelerate convergence [38]. To handle multi-layer neural networks, a tanh activation function is used in the output scaling layer.

TABLE 2. Parameters of DDPG training.

Parameter	Value
Actor network learning rate	0.0001
Critic network learning rate	0.001
Update rate	0.001
Batch size	64
Discount factor	0.99
Experience buffer size	10,000
Sample time	0.05

TABLE 3. Parameters of the system.

Parameter	Value
Vehicle mass, m_i	1600 kg
Moment of inertia, I_i	2875 mN/s^2
Front cornering stiffness coefficient, $C_{\alpha f,i}$	19000 N/m
Rear cornering stiffness coefficient, $C_{\alpha r,i}$	33000 N/m
Distance from cog to front axle, $l_{f,i}$	1.4 m
Distance from cog to rear axle, $l_{r,i}$	1.6 m
Spacing policy standstill distance, r_i	4 m
Spacing policy time-gap, h_i	0.55 s

The critic network consists of two input layers (state and action), one output layer, and three fully connected hidden layers. Both action and state path merge into the output layer through addition layer. The hidden layers have 100 neurons each with ReLU activation function, while the linear activation function is used in the addition layer. To eliminate the dimensional influence between the data, we use batch normalization that can transform the input data to a normal distribution. To improve the efficiency of exploration, the Ornstein-Uhlenbeck process noise is applied with variance 0.6 and 0.1 for acceleration and yaw rate input, respectively. The decay rate of the noise is chosen as 10^{-5}. Other training parameters are as listed in Table 2.

The DDPG-based extended look-ahead algorithm is trained using RL Toolbox in MATLAB. We trained the

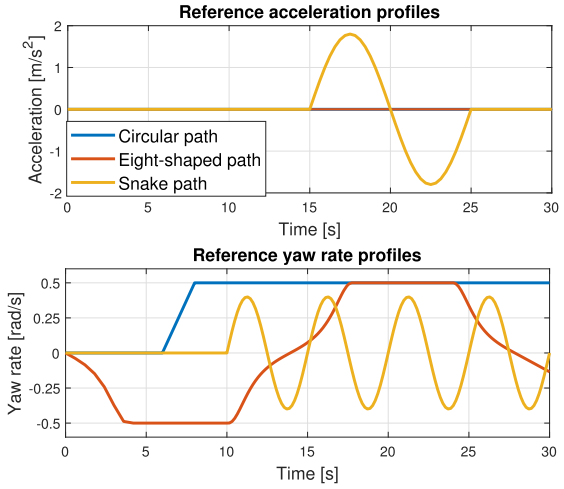

FIGURE 6. Reference acceleration and yaw rate profiles for all scenarios.

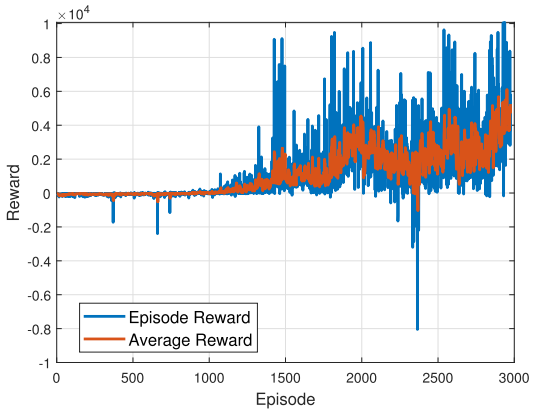

FIGURE 7. Performance of total and average reward versus episode.

algorithm for a platoon with two vehicles, modeled as a nonlinear dynamic single-track vehicle. Vehicle 1 acts as a leader and is directly controlled with predefined acceleration and yaw rate profile. Vehicle 2 acts as a follower vehicle and is controlled by the DDPG-based extended look-ahead algorithm. The parameters for vehicles are as given in Table 3.

B. ALGORITHM TRAINING AND SIMULATION RESULTS

In this subsection, the training result of the DDPG-based extended look-ahead algorithm and the simulations are presented. The platoon consists of leader and follower vehicles which are modeled using a nonlinear dynamic single-track vehicle model as in (2)–(4). The platoon is simulated in three scenarios: the circular path, eight-shaped path, and snake path. The circular path can be considered as the simplest scenario, consisting of a straight path and a circular path with a constant curvature. The eight-shaped path consists of a zero, varying, and constant curvature, where the robustness of the designed controller is evaluated against an aggressive curvature trajectory. The snake path consists of a straight path and a wave path with varying curvature and acceleration. Within this path, the robustness of the controller algorithm is evaluated simultaneously against both varying curvature and acceleration. Fig. 6 shows the acceleration and yaw rate profile of each scenario. In the training process, the leader and the follower vehicle are initiated with the velocity of 24 m/s (equals to 86.4 km/h) and 22 m/s (equals to 79.2 km/h), respectively, and the initial positions of both vehicles and the reference path are randomly changed before the beginning of each episode to prevent over-fitting of the model to a certain scenario, while also ensure the reliability of the agent for all scenarios. In other words, each episode has different initial longitudinal and lateral spacing errors.

The total and last five average reward per episode are as shown in Fig. 7. Training performance is considered good if the reward value is high. As expected, the value of cumulative reward increases with the increase of episode. The designed DDPG-based extended look-ahead algorithm converges after training for 2975 episode with the last five average reward equals to 5251. Due to the random change of scenarios, it can be observed that total reward is declining between episode 2000 and 2500. However, the proposed algorithm shows its robustness against changing scenario after episode 2500.

C. SIMULATION RESULTS

A vehicle platoon typically maneuvers at normal driving conditions, i.e, low acceleration/deceleration, and low yaw rate. To demonstrate the potency of the DDPG-based extended look-ahead controller, we simulated the trained agent for a platoon with 4 vehicles in the three different scenarios: circular path, eight-shaped path, and snake path scenario. The circular path is composed of a straight and circular path to assess the robustness of the designed controller against a delicate curvature changes. The eight-shaped path is generated by half circles and quintic polynomial functions, and is useful to evaluated the robustness of the controller against a varying curvature. The snake path is composed of a sinusoidal yaw rate with varying frequency and varying velocity, and used to demonstrate the robustness of the controller in handling varying curvature and velocity at the same time. It should be noted that the second and third scenario are not typical maneuvers of a vehicle platoon, but they were conducted to evaluate the performance of our designed controller to handle challenging maneuvers.

To model all vehicles, we used a dynamic single-track model, as in (2)–(4). Vehicle 1 acts as a leader and is controlled directly using respective acceleration and yaw rate profile as in Fig. 6. Vehicle i, with $i = \{2, 3, 4\}$, is controlled by the DDPG-based extended look-ahead algorithm with the preceding vehicle $i-1$ as its reference. For all scenarios, vehicle 1 starts at a random initial position $(50 + \Delta x, 0)$ m, $|\Delta x| \leq 2$, with initial orientation $\pi/4$ rad, while vehicle i, with $i = \{2, 3, 4\}$, starts at a random initial position $(50 - 15(i-1) + \Delta x, -5 + \Delta y)$ m, $|\Delta x| \leq 2$, $|\Delta y| \leq 0.5$, with initial orientation 0 rad.

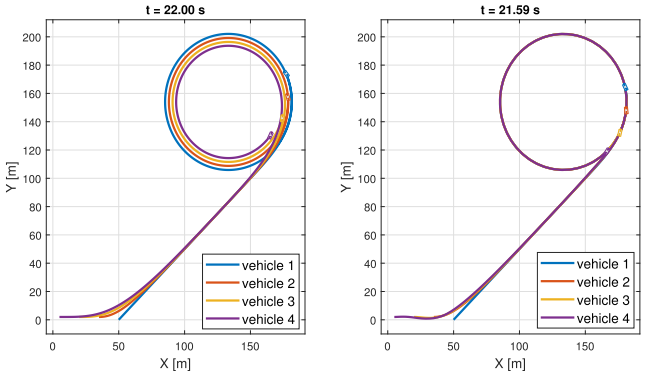

FIGURE 8. Trajectory of vehicles for the circular path scenario: regular look-ahead (left), DDPG-based extended look-ahead (right).

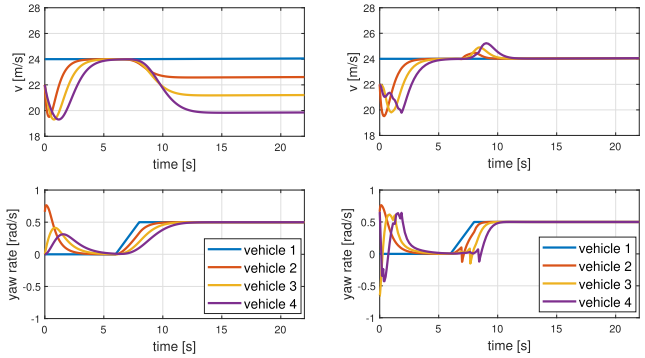

FIGURE 9. Velocity and yaw rate of vehicles for the circular path scenario: regular look-ahead (left), DDPG-based extended look-ahead (right).

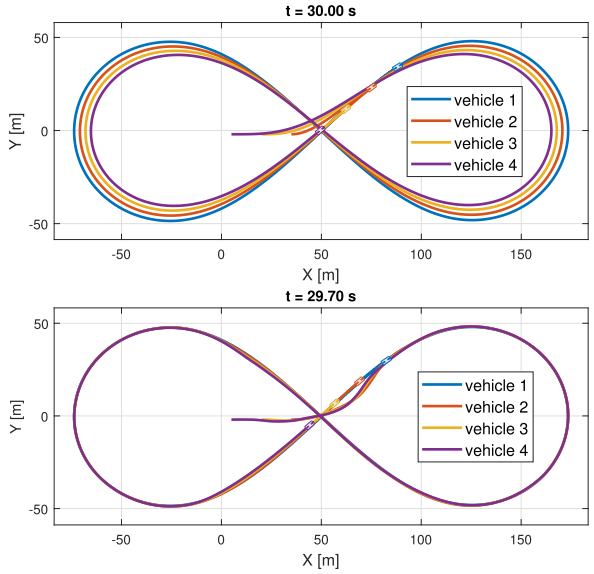

FIGURE 10. Trajectory of vehicles for the eight-shaped path scenario: regular look-ahead (top), DDPG-based extended look-ahead (bottom).

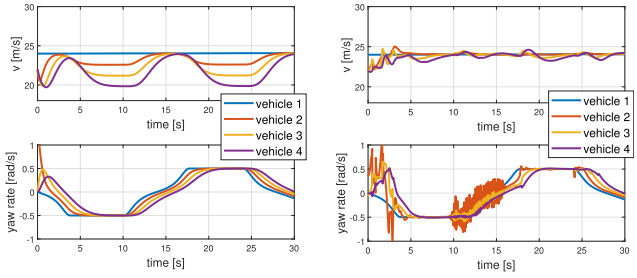

FIGURE 11. Velocity and yaw rate of vehicles for the eight-shaped path scenario: regular look-ahead (left), DDPG-based extended look-ahead (right).

For comparison purposes, we simulated the regular look-ahead controller designed in [7] with identical vehicle parameters and initial conditions with the DDPG-based extended look-ahead controller. This regular look-ahead controller adapts the concept of pure-pursuit controller, in which the follower vehicle does not have any path or trajectory information of its preceding vehicle, and can only track the preceding vehicle by measuring the distance from the front of the follower to the rear of the preceding vehicle.

Fig. 8 shows the trajectory of vehicles for the circular path scenario. The figure on the left shows the trajectory of vehicles using the regular look-ahead controller. As expected, the follower vehicle can only track its preceding vehicle's trajectory on a straight path. Without any path or trajectory information of the preceding vehicle, the follower vehicle suffers cutting-corner, as evidenced by the difference in the path curvature of the follower and its preceding. The difference in curvature can be explained by observing Fig. 9, where all follower vehicles with the regular look-ahead controller have lower turning velocity than the ones with the DDPG-based extended look-ahead controller. On the other hand, as depicted in Fig. 8(right), the follower vehicles with the DDPG-based extended look-ahead controller can track their predecessor's trajectory for the circular path, i.e., the path with constant velocity and varying yaw rate. Fig. 9(right) confirms that all follower vehicles track the velocity and the yaw rate of their respective preceding vehicle with sufficiently small errors.

The overall response of the first scenario is similar to one of the second scenario, the eight-shaped path, as shown in Fig. 10. For the DDPG-based extended look-ahead controller, it can be observed from Fig. 11(right) that the difference between the velocity and the yaw rate of the follower vehicles and vehicle 1 is relatively small, thus ensuring that all vehicles drive on the same path. From both first and second scenario, we can conclude that the proposed DDPG-based extended look-ahead algorithm handles varying yaw rate and constant velocity in a satisfactory manner.

The third scenario is conducted to assess the performance of the design DDPG-based algorithm against both varying yaw rate and velocity. From $t = 0$ s until $t = 10$ s, vehicle 1 drives at a straight path with constant velocity. From $t = 10$ s onward, a sinusoidal yaw rate with magnitude 0.4 rad/s and frequency 0.5 Hz is applied as input to the vehicle. From $t = 15$ s until $t = 25$ s, a one cycle sinusoidal acceleration with magnitude 1.8 m/s^2 is applied. As shown by Fig. 12(right), all follower vehicles with the DDPG-based

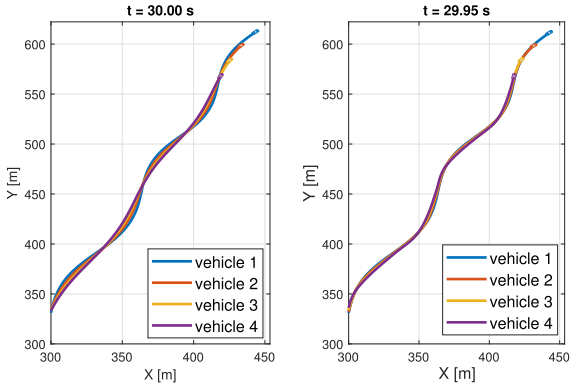

FIGURE 12. Trajectory of vehicles for the snake path scenario: regular look-ahead (left), DDPG-based extended look-ahead (right).

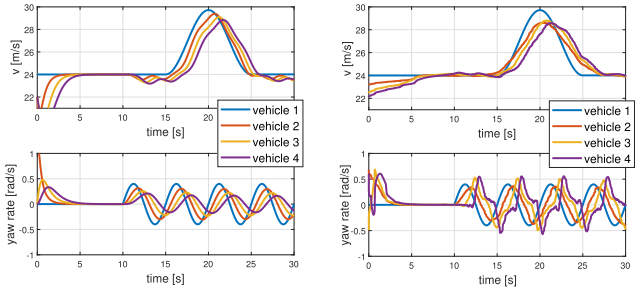

FIGURE 13. Trajectory of vehicles for the snake path scenario: regular look-ahead (left), DDPG-based extended look-ahead (right).

extended look-ahead controller respond quickly to follow their respective preceding vehicle on a varying yaw rate and varying acceleration. The difference between the velocity of the followers and their preceding vehicle from $t = 15$ s onward indicates a varying deceleration (see Fig. 13(right), which implies a varying desired distance due to the constant time-gap spacing policy (1). Therefore, it can be concluded from these three scenarios that our DDPG-based extended look-ahead controller can be applied to multiple vehicles in a platoon, where vehicle $i - 1$ acts as a target tracking for vehicle i, further confirming the scalability of the controller. The simulation results also confirm that the DDPG-based controller can be implemented in a nonlinear dynamic model, which takes coupled longitudinal and lateral dynamics into account, using the designed dynamic input inversion. Moreover, our designed controller is proven to be effective against cutting-corner in the situation where the preceding vehicle's path is unknown, justified by the acceptable error tolerance for various road conditions with varying yaw rate and velocity.

V. EXPERIMENT WITH DONKEY CARS

A. EXPERIMENTAL SETUP

In this section, we describe the experimental setup for the implementation of DDPG-based extended look-ahead controller. Additionally, we present the experimental results to validate theoretical and simulation results. We implemented

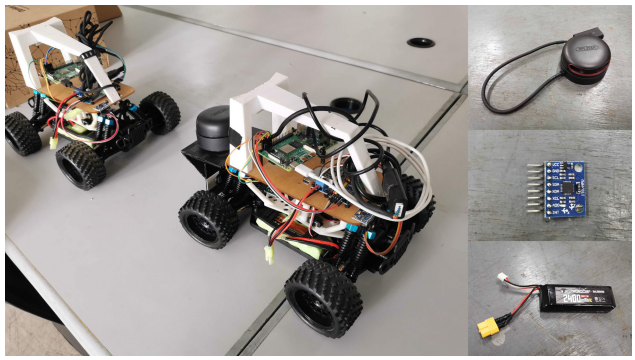

FIGURE 14. The Donkey Car used in simulation. The rear-wheel drive system is throttled using a dc motor and the front wheels are steered using a servo. RPLIDAR-A2, MPU-6050, and additional battery (right side, from top to bottom) are added to the car.

the DDPG-based extended look-ahead controller in a platoon consisting of two Donkey Cars. A Donkey Car is a four-wheel small-scale car with a modular design, and is developed using open source libraries based on Python. A Donkey Car is typically modeled by a kinematic single-track model instead of a nonlinear dynamic single-track model that we used on the simulation By using Donkey Cars, we further verified the generalization capability of our controller to different vehicle models. In this experiment, we used two starter kit Donkey Cars (HSP 94186 model) with servo drivers PCA 9685 to control the throttle and steering, and additional components as follows (see Fig. 14):

- Raspberry Pi 4 Model B: a single-board computer (SBC) that acts as the brain of the car, attached to each leader and follower vehicle. The DDPG-based controller algorithm is embedded into the SBC of the follower vehicle, while the SBC of the leader vehicle controls the throttle and steering and also transmits all necessary signals for the controller through Wi-Fi protocols.
- SLAMTEC RPLIDAR-A2: a small lidar with 0.2 - 16 m measuring range, 16K sampling frequency, and 5 - 10 Hz rotational speed. The lidar is attached to the follower vehicle to measure the inter-vehicle distance.
- InvenSense MPU-6050: a micro electro-mechanical system with a 3-axis gyroscope and 3-axis accelerometer to measure linear and angular velocities and accelerations, attached to each leader and follower vehicle.
- Extra battery: since the original Donkey Car kit uses 1,100 mAh battery, we replace it with a 2,400 mAh battery in each vehicle to power Donkey Car, Raspberry Pi, and lidar.

The DDPG-algorithm that is trained in MATLAB in the previous section is exported to TensorFlow using the built-in function `exportNetworkToTensorFlow`. The function exports the deep learning network and saves it as a TensorFlow model in the Python package.

In order to validate the simulation results, we define a circular trajectory for the leader vehicle. The leader vehicle is directly controlled, while the follower vehicle is controlled by the DDPG-based extended look-ahead algorithm. Since

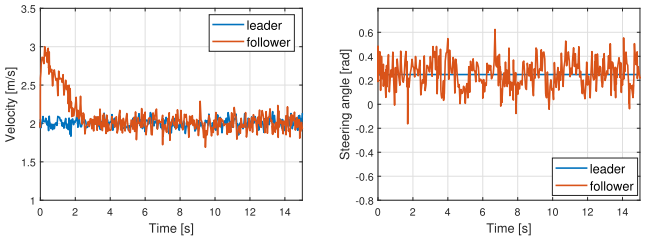

FIGURE 15. Velocity (left) and steering angle (right) of the leader and follower vehicle.

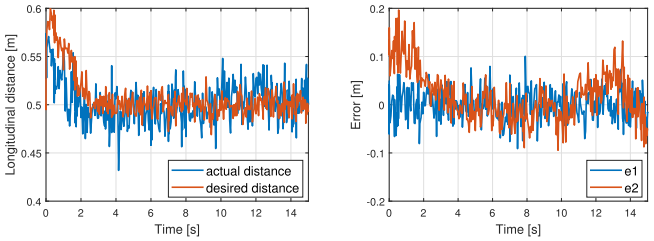

FIGURE 16. Longitudinal distance (left) and errors (right) of the experimental results.

the longitudinal motion of a Donkey Car is controlled by providing a throttle input, we designed a PID controller for the leader vehicle to achieve a constant velocity of 2 m/s. On the other hand, the lateral motion of a Donkey Car is directly-controlled by providing a constant steering angle input of 0.2479 rad. The standstill distance parameter for the spacing policy is chosen as $r_i = 0.3$ m, and the time-gap is chosen as $h_i = 0.1$ s. The lidar is used to measure the inter-vehicle distance and angle of an object relative to the lidar orientation. The Inertial Measurement Unit (IMU) utilizes a low pass filter to reduce noise effect and increase sensor's accuracy. The necessary signals from the leader vehicle (such as, velocity and relative orientation) with a sample rate of 60 Hz are then transmitted to the follower vehicle through a Wi-Fi communication, while the relative inter-vehicle distance is measured using the lidar. The DDPG-based controller then calculates the inputs needed for the follower vehicle.

B. EXPERIMENTAL RESULTS

For the experiment, we placed the follower vehicle behind the leader vehicle with a relatively small initial longitudinal and lateral errors. The orientation of the follower is placed in the same direction as the leader vehicle. This positioning is done to ensure that the follower vehicle moves forward with a small initial orientation error as possible. The leader vehicle maneuvers on a circular path with a constant velocity of 2 m/s and a constant steering angle of 0.2479 rad. The velocity and steering angle of both leader and follower vehicle are shown in Fig. 15. As observed, the velocity convergence of the follower vehicle is apparent after $t = 3$ s, subject to the inter-vehicle longitudinal spacing policy. Fig 16(left) shows the desired longitudinal distance, based on the constant time-gap spacing policy (1), and the actual inter-vehicle distance. It can be observed that the desired distance varies due to the varying velocity of the follower vehicle. During $t = 0$ s until $t = 2$ s, the desired distance is bigger than the actual distance due to a high velocity of the follower vehicle. From $t = 4$ s onward, the actual distance converges to 0.5 m. On the other hand, although the measured steering angle suffers from a bigger noise than the velocity measurement, it is shown that the steering angle of the follower also converges to the steering angle of the leader vehicle. To analyze the runtime efficiency of the proposed algorithm, we measured the execution time in every time step using the built-in clock function in Raspberry Pi. The average execution time is obtained as 0.012 ms. Since the controller agent is already trained in MATLAB, the computational time is relatively fast and does not interfere with the control purpose, which evidently shown by the error plot in Fig. 16(right). However, from the figure, it should be noted that lateral error e_2 is relatively larger than the longitudinal error e_1. The large lateral error might be caused by several factors: the inaccuracy of the inertial measurement unit, delay of the actuators, and the non-deterministic delay in the Wi-Fi communication. Despite of these factors, the average errors are smaller than 0.1 m, which then show satisfactory results of the proposed DDPG-based extended look-ahead algorithm.

C. REMARKS AND DISCUSSION

The experiment with Donkey Cars showed the potential of the DDPG-based extended look-ahead controller implementation, which also can be seen as a first step towards the implementation in real vehicles. It should be noted that although the dynamics of a Donkey Car is different than the one of a real vehicle, in a normal driving condition in a platoon (i.e., low acceleration/deceleration and low yaw rate) a real vehicle can be modeled as a linear dynamic single-track model, or even a kinematic single-track model. To adapt our controller to a real vehicle model, we can design a secondary controller that controls the acceleration or speed, instead of throttle in Donkey Car, internally compensating for the vehicle parameters.

In the simulation, it is assumed that all vehicles can measure the inter-vehicle distance accurately, and no delays are involved in wireless communication. However, it should be noted that the experiment with Donkey Cars and future implementation of the DDPG-based extended look-ahead controller in real vehicles have to take sensors' accuracy, reliability, and potential failures into account. In [39], a failure-resilient platooning system is proposed to handle potential failures of sensors. In this system, the failed sensors is emulated by collectively utilizing other sensors in the platoon. The control system then can instantaneously reconfigured the cooperative mode using only the live sensors. Authors in [40] proposed levels of failure with immediate stop of the vehicle to address sensor malfunctions. To address the problem where some signals cannot be accurately measured or are unavailable due to sensors malfunction, an observer-based

secure control as proposed in [21] can be adapted. In the experiment with Donkey Cars, we used IEEE.802.11 wireless protocol which works well for our controller. On the other hand, the Vehicle-to-Everything (V2X) communication uses ETSI ITS-G5 and DSRC/WAVE technology based on IEEE 802.11p, which is a more robust protocol against fading and multipath propagation effects of signals in a vehicular environment. In the situation where the communication is delayed or even failed, we can adapt a failure tolerance system as the one designed in [41]. To address a deliberate manipulation of the system through cyber-attacks, authors in [42] proposed a reliable trust-based platoon service. To handle the aforementioned sensors failures, our DDPG-based controller can be used in conjunction with other controllers or systems designed in those studies.

VI. CONCLUSION

In this paper, we propose a Deep Deterministic Policy Gradient (DDPG) algorithm with extended look-ahead approach for the control of longitudinal and lateral vehicle platooning. This method combines the ability of DDPG to handle nonlinear system and uncertainties with the advantage of cost-effectiveness and feasibility of the look-ahead approach. Without the information of reference paths or lane markings, the application of the look-ahead approach can cause cutting-corner phenomenon. The cutting-corners escalate upstream in the platoon, thus affecting the lateral tracking performance of all vehicles in the platoon. Moreover, a nonlinear dynamics vehicle model must be considered for dynamic longitudinal and lateral maneuvers. To solve the cutting-corner problem, we adapted the extended look-ahead approach that redefined the tracking objective point into a nonlinear dynamic vehicle model. With an input inversion, the original input of the nonlinear dynamic vehicle model can be obtained from acceleration and steering angle input from DDPG algorithm. The effectiveness of the proposed DDPG-based extended look-ahead approach is evaluated in both simulation and experimental environments. The results demonstrate that the DDPG-based extended look-ahead algorithm can meet the requirements of longitudinal and lateral tracking by reducing cutting-corners under different road, velocity, and yaw rate conditions.

The future work would be concentrated on the implementation of the algorithm into a platoon with more than two vehicles, and on the design optimization of the DDPG algorithm to improve the performance, error minimization, and convergence speed. Moreover, the following research can incorporate the string stability analysis of the platoon and can focus on the implementation on real vehicles to further verify the reliability of the DDPG-based extended look-ahead algorithm.

REFERENCES

[1] A. Geiger, M. Lauer, F. Moosmann, B. Ranft, H. Rapp, C. Stiller, and J. Ziegler, "Team AnnieWAY's entry to the 2011 grand cooperative driving challenge," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 3, pp. 1008–1017, Sep. 2012.

[2] R. Rajamani, *Vehicle Dynamics and Control* (Mechanical Engineering Series). Boston, MA, USA: Springer, 2012.

[3] J. Ploeg, A. F. A. Serrarens, and G. J. Heijenk, "Connect & drive: Design and evaluation of cooperative adaptive cruise control for congestion reduction," *J. Mod. Transp.*, vol. 19, no. 3, pp. 207–213, Sep. 2011.

[4] H. M. Fahmy, M. A. A. E. Ghany, and G. Baumann, "Vehicle risk assessment and control for lane-keeping and collision avoidance at low-speed and high-speed scenarios," *IEEE Trans. Veh. Technol.*, vol. 67, no. 6, pp. 4806–4818, Jun. 2018.

[5] R. He, H. Lv, S. Zhang, D. Zhang, and H. Zhang, "Lane following method based on improved DDPG algorithm," *Sensors*, vol. 21, no. 14, p. 4827, Jul. 2021.

[6] J. Ploeg, B. T. M. Scheepers, E. van Nunen, N. van de Wouw, and H. Nijmeijer, "Design and experimental evaluation of cooperative adaptive cruise control," in *Proc. 14th Int. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Washington, DC, USA, Oct. 2011, pp. 260–265.

[7] A. Bayuwindra, J. Ploeg, E. Lefeber, and H. Nijmeijer, "Combined longitudinal and lateral control of car-like vehicle platooning with extended look-ahead," *IEEE Trans. Control Syst. Technol.*, vol. 28, no. 3, pp. 790–803, May 2020.

[8] A. Bayuwindra, E. Lefeber, J. Ploeg, and H. Nijmeijer, "Extended look-ahead tracking controller with orientation-error observer for vehicle platooning," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 11, pp. 4808–4821, Nov. 2020.

[9] S. Wei, Y. Zou, X. Zhang, T. Zhang, and X. Li, "An integrated longitudinal and lateral vehicle following control system with radar and vehicle-to-vehicle communication," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1116–1127, Feb. 2019.

[10] Y. Wang, N. Bian, L. Zhang, and H. Chen, "Coordinated lateral and longitudinal vehicle-following control of connected and automated vehicles considering nonlinear dynamics," *IEEE Control Syst. Lett.*, vol. 4, no. 4, pp. 1054–1059, Oct. 2020.

[11] H. T. Trinh, S.-H. Bae, and D. Q. Tran, "Deep reinforcement learning for vehicle platooning at a signalized intersection in mixed traffic with partial detection," *Appl. Sci.*, vol. 12, no. 19, p. 10145, Oct. 2022.

[12] P. Qin, H. Tan, H. Li, and X. Wen, "Deep reinforcement learning car-following model considering longitudinal and lateral control," *Sustainability*, vol. 14, no. 24, p. 16705, Dec. 2022.

[13] K. Berns and R. Dillmann, "A neural network approach for the control of a tracking behavior," in *Proc. 5th Int. Conf. Adv. Robot. 'Robots Unstructured Environ.*, vol. 1, Jun. 1991, pp. 500–503.

[14] K. Berns, R. Dillmann, and U. Zachmann, "Reinforcement-learning for the control of an autonomous mobile robot," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, vol. 3, Jul. 1992, pp. 1808–1815.

[15] X. Dai, C. K. Li, and A. B. Rad, "An approach to tune fuzzy controllers based on reinforcement learning for autonomous vehicle control," *IEEE Trans. Intell. Transp. Syst.*, vol. 6, no. 3, pp. 285–293, Sep. 2005.

[16] Z. Huang, X. Xu, H. He, J. Tan, and Z. Sun, "Parameterized batch reinforcement learning for longitudinal control of autonomous land vehicles," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 49, no. 4, pp. 730–741, Apr. 2019.

[17] H. Chae, C. M. Kang, B. Kim, J. Kim, C. C. Chung, and J. W. Choi, "Autonomous braking system via deep reinforcement learning," in *Proc. IEEE 20th Int. Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2017, pp. 1–6.

[18] W. Xia, H. Li, and B. Li, "A control strategy of autonomous vehicles based on deep reinforcement learning," in *Proc. 9th Int. Symp. Comput. Intell. Design (ISCID)*, vol. 2, Dec. 2016, pp. 198–201.

[19] L. Cui, K. Ozbay, and Z.-P. Jiang, "Combined longitudinal and lateral control of autonomous vehicles based on reinforcement learning," in *Proc. Amer. Control Conf. (ACC)*, New Orleans, LA, USA, May 2021, pp. 1929–1934.

[20] T. Sogabe, D. B. Malla, S. Takayama, S. Shin, K. Sakamoto, K. Yamaguchi, T. P. Singh, M. Sogabe, T. Hirata, and Y. Okada, "Smart grid optimization by deep reinforcement learning over discrete and continuous action space," in *Proc. IEEE 7th World Conf. Photovoltaic Energy Convers. (WCPEC), Joint Conf. 45th IEEE PVSC, 28th PVSEC 34th EU PVSEC*, Jun. 2018, pp. 3794–3796.

[21] S. Khodadadi, T. K. Tasooji, and H. J. Marquez, "Observer-based secure control for vehicular platooning under DoS attacks," *IEEE Access*, vol. 11, pp. 20542–20552, 2023.

[22] S. Lu, Y. Cai, L. Chen, H. Wang, X. Sun, and Y. Jia, "A sharing deep reinforcement learning method for efficient vehicle platooning control," *IET Intell. Transp. Syst.*, vol. 16, no. 12, pp. 1697–1709, Dec. 2022, doi: 10.1049/itr2.12120.
[23] J. Yang, W. Peng, and C. Sun, "A learning control method of automated vehicle platoon at straight path with DDPG-based PID," *Electronics*, vol. 10, no. 21, p. 2580, Oct. 2021.
[24] S. B. Prathiba, G. Raja, K. Dev, N. Kumar, and M. Guizani, "A hybrid deep reinforcement learning for autonomous vehicles smart-platooning," *IEEE Trans. Veh. Technol.*, vol. 70, no. 12, pp. 13340–13350, Dec. 2021.
[25] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.
[26] F. Guan, H. Xu, and Y. Tian, "Evaluation of roadside LiDAR-based and vision-based multi-model all-traffic trajectory data," *Sensors*, vol. 23, no. 12, p. 5377, Jun. 2023.
[27] D. Swaroop, J. K. Hedrick, C. C. Chien, and P. Ioannou, "A comparision of spacing and headway control laws for automatically controlled vehicles," *Vehicle Syst. Dyn.*, vol. 23, no. 1, pp. 597–625, Jan. 1994.
[28] L. Silverman, "Inversion of multivariable linear systems," *IEEE Trans. Autom. Control*, vol. AC-14, no. 3, pp. 270–276, Jun. 1969.
[29] H. Khalil, *Nonlinear Systems*. London, U.K.: Pearson Education, 2002.
[30] A. Isidori, *Nonlinear Control Systems* (Communications and Control Engineering). London, U.K.: Springer, 1995.
[31] R. Attia, R. Orjuela, and M. Basset, "Coupled longitudinal and lateral control strategy improving lateral stability for autonomous vehicle," in *Proc. Amer. Control Conf. (ACC)*, Montreal, QC, Canada, Jun. 2012, pp. 6509–6514.
[32] R. Rajamani and C. Zhu, "Semi-autonomous adaptive cruise control systems," *IEEE Trans. Veh. Technol.*, vol. 51, no. 5, pp. 1186–1192, Sep. 2002.
[33] A. Farag, O. M. Abdulaaty, A. Hussein, and O. M. Shehata, "Reinforcement learning based approach for multi-vehicle platooning problem with nonlinear dynamic behavior," in *Proc. 34th Conf. Neural Inf. Process. Syst.*, Dec. 2020, pp. 1–10.
[34] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. Cambridge, MA, USA: Bradford Books, Nov. 2018.
[35] S. Choi, T. Le, Q. Nguyen, M. Layek, S. Lee, and T. Chung, "Toward self-driving bicycles using state-of-the-art deep reinforcement learning algorithms," *Symmetry*, vol. 11, no. 2, p. 290, Feb. 2019.
[36] X. Luo, T. Chen, M. Li, and S. Li, "Platoon control of automatic vehicles based on deep deterministic policy gradient," in *Proc. 40th Chin. Control Conf. (CCC)*, Jul. 2021, pp. 6154–6159.
[37] A. Karalakou, D. Troullinos, G. Chalkiadakis, and M. Papageorgiou, "Deep reinforcement learning reward function design for autonomous driving in lane-free traffic," *Systems*, vol. 11, no. 3, p. 134, Mar. 2023.
[38] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. 30th Int. Conf. Mach. Learn.*, Atlanta, GA, USA, 2013, pp. 1–6.
[39] C. Koo, J. Park, T. Ahn, H. Kim, J.-C. Kim, and Y. Eun, "Phalanx: Failure-resilient truck platooning system," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Apr. 2023, pp. 1–6.
[40] J. Lygeros, D. N. Godbole, and M. Broucke, "A fault tolerant control architecture for automated highway systems," *IEEE Trans. Control Syst. Technol.*, vol. 8, no. 2, pp. 205–219, Mar. 2000.
[41] J. Ploeg, N. V. D. Wouw, and H. Nijmeijer, "Fault tolerance of cooperative vehicle platoons subject to communication delay," *IFAC-PapersOnLine*, vol. 48, no. 12, pp. 352–357, Jan. 2015.
[42] H. Hu, R. Lu, Z. Zhang, and J. Shao, "REPLACE: A reliable trust-based platoon service recommendation scheme in VANET," *IEEE Trans. Veh. Technol.*, vol. 66, no. 2, pp. 1786–1797, Feb. 2017.

ANGGERA BAYUWINDRA was born in Bandung, Indonesia, in 1984. He received the B.Sc. and M.Sc. degrees in electrical engineering (specializing in control and intelligent system) from the Bandung Institute of Technology, Bandung, in 2006 and 2012, respectively, and the Ph.D. degree in dynamics and control, mechanical engineering from the Eindhoven University of Technology, Eindhoven, The Netherlands, in 2019.

From 2006 to 2008, he was an Engineer with Infineon Technologies, Batam, Indonesia. From 2021 to 2022, he was a University Lecturer with the Eindhoven University of Technology, where he was involved in teaching and research related to vehicle dynamics, nonlinear control, and control of mechanical systems. Since 2022, he has been a Lecturer with the School of Electrical Engineering and Informatics, Bandung Institute of Technology. His research interests include control system design, robotics, intelligent transportation systems (ITS), and cooperative and autonomous vehicle.

LEON WONOHITO received the bachelor's degree in science from Gadjah Mada University, in 2020. He is currently pursuing the master's degree in smart control system with the Bandung Institute of Technology, Bandung, Indonesia. His research interests include deep learning, especially in control application and the Internet of Things (IoT).

BAMBANG R. TRILAKSONO (Member, IEEE) was born in Banyuwangi, Indonesia, in 1962. He received the bachelor's degree in electrical engineering from the Bandung Institute of Technology, Bandung, Indonesia, in 1986, and the master's and Ph.D. degrees in electrical engineering from Waseda University, Tokyo, Japan, in 1991 and 1994, respectively.

He is currently a Professor with the School of Electrical Engineering and Informatics, Bandung Institute of Technology. He is also a Research Fellow with the University of New South Wales, Sydney, NSW, Australia. His research interests include control systems, artificial intelligence, and robotics.

Prof. Trilaksono serves as an editorial board member for several journals. He was a recipient of several awards, including the Toray Science and Technology Award, in 2004. He served as the Vice Rector for Research, Innovation and Partnerships, ITB, 2015–2019.