2022 18th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob) | 978-1-6654-6975-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/WIMOB55322.2022.9941534

Drone-Assisted Lane Change Maneuver using Reinforcement Learning with Dynamic Reward Function

Jialin Hao*, Rola Naja†‡, Djamal Zeghlache*
*Télécom SudParis, Institut Polytechnique de Paris, Palaiseau, France
†ECE Paris Research Center
‡Li-Parad Laboratory, University of Versailles Saint Quentin

{jialin.hao, djamal.zeghlache}@telecom-sudparis.eu
{rola.naja}@ul.edu.lb

***Abstract*—This paper provides a Lane Change Assistance (LCA) platform that communicates with Unmanned Aerial Vehicles (UAV). The proposed platform is based on a reinforcement learning technique, where a Deep Q-Network (DQN) is trained to make lane change decisions. The reward function of the DQN agent considers safety, comfort and efficiency perspectives. Specifically, the safety reward, based on the road vehicular density, is adapted dynamically by the drone during the training phase. Performance analysis proves that the proposed platform improves the total travel time while reducing the collision rate and responding to urgent lane changes in a timely manner.**

***Index Terms*—lane change, reinforcement learning, dynamic reward function**

I. Introduction

Lane changing is one fundamental and crucial function expected to be embedded in a smart vehicle. The lane changing maneuver can be a demanding task since the vehicle needs to alertly watch the leading vehicle on its lane and surrounding vehicles on the target lane, and to perform proper actions according to the potential adversarial or cooperative reactions demonstrated by those relevant vehicles [1].

Research on automated lane changing maneuvers has been extensively conducted, and the work can broadly be divided into two functional categories: a decision-making module and a control execution module. A decision-making module can be viewed as a strategic or tactical level function which issues a lane change command according to a selected route (e.g. exiting the highway from an off-ramp) or a desired driving condition (e.g. passing a slow vehicle in front).

When a lane change command is given, the lane changing vehicle performs operational control to coordinate the longitudinal and lateral movements for a safe, smooth and efficient lane change maneuver.

Automated lane changing functions built on rule-based models may perform well under pre-defined operating conditions, but they may be prone to failure when unexpected situations are encountered. Therefore, in our study, we propose to adopt a Deep Reinforcement Learning (DRL) based approach, and more specifically a Deep Q-Network (DQN) in order to train the vehicle agent, called the ego vehicle. The latter learns an automated lane change behavior such that it can intelligently make a lane change under unexpected scenarios. Particularly, we treat both state space and action space as continuous, and design a unique Q-function approximator to estimate the total reward which is an accumulated reward over a lane changing process. The state space is retrieved by data sets stored at the ego vehicle.

Lane change problem has attracted numerous researchers who oriented their efforts towards enhancing lane change intention prediction, efficiency, collision rate, and overall reward over the highway. In [2], authors address the lane change intention prediction problem with an online approach based on Support Vector Machine and Bayesian filtering. With this approach, they are able to predict driver intention to change lanes on average 1.3 seconds in advance, with a maximum prediction horizon of 3.29 seconds. In [3], the lane change intention prediction problem is formulated by labeling unlabeled data using a limited number of features. In this way, the process of manual labeling of training dataset is released.

Authors in [4–10] model the lane change system as a Markov decision process but with different state spaces, action spaces and reward functions. In [4], a DQN agent is trained with the Deep Deterministic Policy Gradient (DDPG) algorithm in order to make lane change decisions and avoid collisions. Moreover, authors take into account the update delay of the remote vehicle. The result shows that the agent learns to make successful lane changes with both lateral and longitudinal control. Nevertheless, the simulation scenario considers a single remote vehicle. [5] and [6] train a hierarchical DQN structure to learn both the lane change decision-making (high-level control) and lane change trajectory planning (low-level control). The reward function is defined with the ego vehicle's yaw rate, yaw acceleration and lane changing time in order to learn a smooth and efficient lane change

This Ph.D program is funded by the DigiCosme project: ANR11LABEX0045DIGICOSME

behavior. In the architecture proposed in [7], the lane change decision is made by maximizing the ego car's acceleration. The input of the trained DQN is a vector of 27 elements, representing the ego vehicle and its 8 neighbors' states. The action space contains 6 elements: stay in the current lane with 4 different accelerations, or change lane to the right or to the left. [8] proposes a DQN based method using grid-form state representation. The longitudinal speed is controlled by a low-level rule-based trajectory controller, the lateral lane change decisions are made by a high-level lateral DQN decision-maker. It is noteworthy that 12 interfering vehicles are considered. However, the research study lacks a collision performance evaluation and a safety guarantee assessment. Authors in [9] adopt an attention-based DRL to address the interaction with surrounding vehicles while making lane change decision. The inputs are images extracted from the car's front view. The algorithm performance evaluation under different realistic scenarios reveals that the proposed DRL algorithm is capable of learning both lane change decision and path planning with 73.5% of successful episodes. [10] considers the overall traffic efficiency instead of the travel efficiency of an individual vehicle. A convolutional DQN is trained with the input consisting of the traffic snapshots.

The majority of the research studies adopted the deep learning algorithms in order to address the lane change problem while reducing the collision rate. Nevertheless, most of these models design the reward function based on local information related to instant speed, direction, the ego vehicle position and its neighbors. That is to say, the existing proposed models lack of a global view of the vehicles states. Therefore, we propose to integrate the drones, or UAVs, that cooperate efficiently with the vehicles owing to their Line-of-Sight links. Furthermore, with drones' high dynamic deployment ability, they can be deployed to required locations to improve communication quality in imperfect communication conditions. More specifically, drones are equipped with dedicated sensors and communication devices, that enable them to undertake various services such as low altitude surveillance, post-disaster rescue and logistics application communication assistance[11]. The cooperation with vehicles' On-Board Units (OBU) and infrastructures allows drones to improve vehicle-to-vehicle connectivity, infrastructure coverage and data collection ability.

In this paper, we present an innovative Lane Change Assistance (LCA) platform based on deep reinforcement learning with the assistance of UAVs. More specifically, we design a dynamic DQN agent reward function, based on global information retrieved from drones. Compared to the existing literature, our contributions are four-fold:

1) We provide timely and accurate acquisition of traffic flow information of the overall highway with the drones' assistance.
2) We devise a lane change decision-making module that considers roads prone to accidents with aggressive surrounding vehicles that induce collisions and ambulance vehicles that force a lane change.
3) We design a comprehensive reward function that integrates comfort, travel efficiency and road active safety. More specifically, the safety function is adapted dynamically by the drones that have a global view of the vehicular network.
4) The lane change decision reduces the total travel time (i.e. maximizes the instant speed) while avoiding collision with surrounding vehicles.

The paper is organized as follows: In the second section, we shed the light on our LCA platform along with its modules. In the third section, we present the adopted DEAR agent (DEep Q-network with the dynAmic Reward function). In the fourth section, we provide a performance analysis before concluding the paper.

II. LCA PLATFORM

A. Lane Change Adopted Scenario

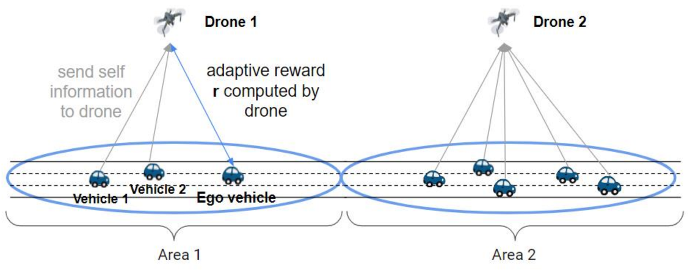
Drone 1
Drone 2
send self information to drone
adaptive reward r computed by drone
Vehicle 1
Vehicle 2
Ego vehicle
Area 1
Area 2

Fig. 1. Highway architecture

We adopt a highway architecture illustrated in Fig. 1. Drones hover over the highway, communicate with vehicles' OBUs over Vehicle-to-Drone (V2D) communications and provide a global view of all the vehicles on the road. Each vehicle broadcasts periodically a Hello packet containing its self information. On the other hand, the drone sends an adaptive parameter r computed from the global information to the ego vehicle in order to assist its lane change process. Each drone is responsible for an area, which size is determined by the drone's communication range.

On the highway, several vehicles move with different speeds, as illustrated in Fig. 2. We assume that the ego vehicle requests to change the lane and move to the desired position on the target lane. The inter-vehicle distance can be calculated from the vehicle's position information. The distance between the desired position and its current position can be calculated from the lane width and its heading angle θ. The lane change aims to reduce this distance while respecting the speed constraint: the maximum speed should not exceed the maximum allowed speed, v_{max} on the road (defined as 100 km/h on our highway scenario).

B. LCA Modules

Our project strives to achieve a safe lane change maneuver while executing two main modules that operate in tandem, as illustrated in Fig. 3.

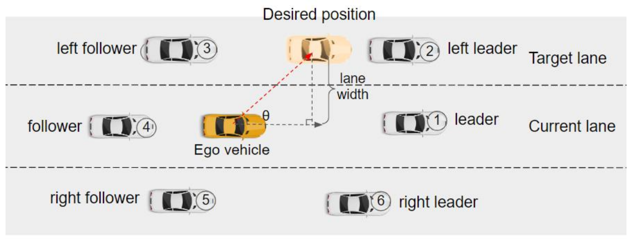

Fig. 2. Ego vehicle surrounded by six neighbors, trying to change lane to the desired position

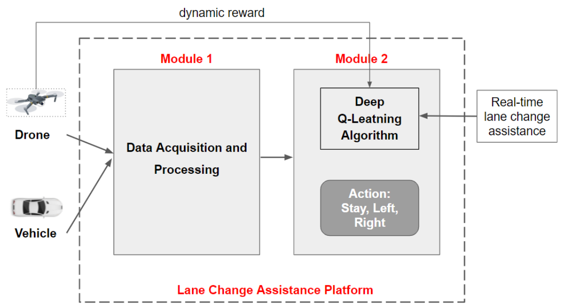
dynamic reward
Module 1
Module 2
Drone
Data Acquisition and Processing
Deep Q-Leatning Algorithm
Real-time lane change assistance
Vehicle
Action: Stay, Left, Right
Lane Change Assistance Platform

Fig. 3. Drone-assisted LCA platform

1) Module 1-Data File acquisition and Processing: Data file acquisition and processing is the building block of our study. In fact, in the pre-lane change phase, data is acquired and collected by the ego vehicle. The collected data file stores kinematic parameters, i.e. GPS coordinates and acceleration that are retrieved by vehicle sensors as well as the overall road vehicular density retrieved by the drones; the computation of the vehicular density will be detailed in III-B3. Afterwards, data file needs to be trimmed in the pre-training phase, as it contains some irrelevant information for lane change decision. Therefore, data processing should reduce the number of variables of a data set in order to provide insightful data analysis. Finally, processed data is fed into LCA module.

2) Module 2- Real time lane change assistance: This module devises lane change maneuver based on a DQN that efficiently assists the driver to take the real-time decision of changing the lane at the optimal instant of time. The deep reinforcement algorithm is described in the following section. It is to be noted that the prediction of a safe lane change should be tightly coupled with data file acquisition and processing module (module 1). Indeed, timely and accurate acquisition of data traffic flow information is a critical element of our project since it lays out the foundation for accident prediction and accurate safe lane change decision.

III. Deep Reinforcement Learning Methodology

A. Q-learning and Deep Q-Network

Q-learning is a reinforcement learning algorithm where the agent tries to learn the optimal action value function $Q^*(s,a)$, defined as the maximum total reward, called the Q-value. When being in a state, s, Q-value takes an action, a, and follows the optimal policy, π^*.

$$Q^*(s,a) = R(s,a) + \gamma \underset{a}{max} Q(s',a), \tag{1}$$

Equation (1) computes the optimal Q-value, where $R(s,a)$ denotes the reward when being in state s and taking action a, s' the next state by taking action a and γ, a factor controlling the contribution of the rewards in the future.

In case the values of $Q(s',a)$ are known, the optimal policy is then to select the action a' that maximizes the expected value of $Q(s',a)$. Further, $Q(s',a)$ depends on $Q(s'',a)$ which will then have a coefficient of γ^2. Thus, the Q-value depends on Q-values of future states, as shown in the following equations:

$$Q(s,a) \rightarrow \gamma Q(s',a) + \gamma^2 Q(s'',a) + \cdots + \gamma^n Q(s'^{\cdots n},a) \tag{2}$$

Mathematically, the state-action value function can be defined as:

$$\begin{aligned} Q_\pi(s,a) &\doteq \mathbb{E}_\pi \left[R_t | S_t = s, A_t = a\right] \\ &= \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s, A_t = a\right] \end{aligned} \tag{3}$$

where S_t is the state at time t, A_t the action taken at time t, α the learning rate and π the policy followed. In practical situations, we implement the optimal value function as follows:

$$\begin{aligned} Q(S_t,A_t) \leftarrow & Q(S_t,A_t) \\ & + \alpha \left[R_{t+1} + \gamma \underset{a}{max} Q(S_{t+1},a) - Q(S_t,A_t)\right], \end{aligned} \tag{4}$$

In Deep Q-Learning, a deep neural network with weights θ is used to approximate the Q-value function, i.e. $Q(s,a;\theta) \approx Q^*(s,a)$. The state is given as the input and the Q-value of all possible actions is generated as the output. The loss function is the mean-squared error between the predicted Q-value and the target Q-value, the latter is defined as $Q^* = R_{t+1} + \gamma \underset{a}{max} Q(S_{t+1},a)$. The network is then trained using stochastic gradient descent by minimizing the loss[12].

The loss function at iteration i is defined as follows:

$$L_i(\theta_i) = \mathbb{E}_M \left[(r + \gamma \underset{a'}{max} Q(s',a';\theta_i^-) - Q(s,a;\theta_i))^2\right] \tag{5}$$

Here, r is the accumulated reward of previous $i-1$ iterations, θ_i^- are the network parameters used to calculate the target at iteration i.

To make the learning process more stable, we adopt two networks with the same architecture during the training process, one called the prediction network, the other called the target network. The parameters of the target network are held fixed for a number of iterations and then periodically updated with the latest version of the trained parameters of the prediction network[13].

B. DQN Lane Change Agent

1) State Space: The agent to be trained is a three-layer DQN with 64, 128, 64 hidden nodes on the first, second, and the third hidden layer. The output is the lane change decision: change lane to the right, change lane to the left, or stay at the current lane. It should be noted that, in this paper, we take into account the 6 possible one-hop neighbors of the ego vehicle, namely the leader, left leader, right leader, follower, left follower and right leader, as illustrated in Fig. 2, when making the lane change decision. Thus, the state space at time step j consists of kinematic parameters of ego vehicle and the 6 neighbors. It can be expressed as

$$o_j = \{o_{ego,j}, o_{1,j}, \cdots, o_{6,j}\}$$

Where $o_{ego,j} = \{risk, x_{ego,j}, y_{ego,j}, v_{ego,j}, a_{ego,j}, l_{ego,j}\}$ is the 6 parameters for the ego vehicle at time j: risk label of current lane $risk$ (0 means no risk detected on current lane, 1 means there is a risk), horizontal position x, vertical position y, longitudinal speed v, acceleration a and current lane id l of the ego vehicle. And $o_{i,j} = \{x_{i,j}, y_{i,j}, v_{i,j}, a_{i,j}, l_{i,j}, d_{i,j}, p_{i,j}\}, i \in (0, 6]$ is the 7 parameters of the i-th neighbor at time j, representing the horizontal position x, vertical position y, longitudinal speed v, acceleration a, current lane id l, distance to the ego vehicle d and vehicle priority p (0 for normal vehicles and 1 for emergency vehicles such as ambulances and police cars). In consequence, the state space is a vector of 48 parameters.

2) Action Space: The action space is defined as $a = \{0, 1, 2\}$ where *0* refers to staying in the current lane, *1* indicates performing a lane change to the right and *2* denotes performing a lane change to the left.

3) Reward Function: The reward function is designed from three different perspectives: comfort (to avoid hard brakes and sharp accelerations), efficiency (reduce the total travel time, prevent unnecessary and frequent lane changes) and safety (avoid collisions)[14]. Consequently, the total reward is the sum of the three rewards:

$$R = R_{comf} + R_{eff} + R_{safe} \tag{6}$$

where R_{comf} is the comfort reward, R_{eff} the efficiency reward and R_{safe} the safety reward.

- *Comfort reward:* The comfort reward is negatively correlated to the fluctuation of acceleration, as shown in (7):

$$R_{comf} = -w_{comf} \cdot \dot{a}_x^2 \tag{7}$$

  $\dot{a}_x$ is the acceleration jitter computed from two adjacent steps, and w_{comf} is the coefficient controlling the comfort weight.

- *Efficiency reward:* The efficiency reward consists of three rewards: position reward, speed reward and lane change reward as indicated in the following equation:

$$R_{eff} = w_{eff}(R_{pos} + R_v + R_{change}) \tag{8}$$

  The position reward, R_{pos} is represented by the distance separating the desired position and current position, as shown in Fig. 2. It can be calculated as $R_{pos} = -\frac{d_{lane}}{sin\theta}$, where d_{lane} is the lane width. The point is that the further the distance is, the lower reward should be assigned. The speed reward, R_v is defined as $R_v = -|v_{max} - v|$. In fact, the closer the speed to the maximum allowed speed is, the higher the reward is.
  The lane change reward, R_{change} is defined such that it penalizes each time a lane change occurs, where α is a constant.

$$R_{change} = \begin{cases} -\alpha, \text{ if change lane} \\ \alpha, \text{ if stay in lane} \end{cases}$$

- *Safety reward:* The safety reward is the sum of collision reward, vehicular density reward, risky reward and blocking reward:

$$R_{safe} = R_{colli} + R_{den} + R_{risk} + R_{block}$$

  The collision reward $R_{colli} = r$ is a dynamic value computed by the drone processor and broadcasted to the vehicles periodically. At the end of an epoch time, the drone calibrates r according to the following rules: whenever a collision occurs, r is decreased in order to penalize the collision; in case no collision occurs in a certain time window, r is increased with the purpose of encouraging the agent performing lane change. The density reward is inversely proportional to the number of vehicle in the current area: $R_{den} = -n_v$, where n_v is the number of vehicles. In fact, the risk of collision will become higher with the vehicular density increasing. Meanwhile, the reward function will decrease.
  The traffic density in the reward function is calculated by collecting the *Hello* packets sent by vehicles. To estimate the traffic density, we divide the highway into fixed size zones that equal to the communication range of 1 kilometer. Each area is covered by a drone which collects the periodical Hello packets from vehicles in the area. Scenario of Fig. 1 illustrates the traffic density calculation scenario, where the drones are located on the highway and collect the packets from vehicles.
  The risky reward R_{risk} and the blocking reward R_{block} are related to the total time of the ego vehicle driving on the risky lane and in front of an emergency vehicle, respectively.

IV. Simulation and performance Results

A. Simulation Setup

In order to validate our platform, we conducted extensive simulation batches with SUMO[15]. The V2D communication is considered perfect owing to drones' Line-of-Sight links and their dynamic deployment ability. As shown in Fig. 4, we consider a 4 km circular highway where the ego vehicle learns to perform a collision-less lane change maneuver. Four drones hover over the highway with a communication area of 1km: they are responsible for estimating the vehicular density in their area.

a) Vehicle mobility and lane change model: Vehicles move according to the Krauss mobility model, where the maximum velocity is 100 km/h. For a more authentic scenario, some vehicles are set to be "aggressive" with impolite behaviors such as low intention to cooperate with others, stay in the leftmost lane for a long time and exceed the speed limit. Moreover, some vehicles are ambulances that lead the in front vehicles to initiate lane change. It is noteworthy, the lane change model adopted is LC2013, which discriminates 4 kinds of lane change: (1) strategic lane change in order to continue the trip, (2) cooperative lane change in order to allow others to change, (3) speed gain lane change which allows for faster speed and (4) obligation to drive on the right for emergency such as when an ambulance or police car pass by [15].

b) Risk Modeling: A random risk will be detected at the beginning of each epoch, such as a car accident in front of the ego vehicle, or some construction work. The risk is distributed uniformly between the 3 lanes and is modelled to happen at one lane at a time. Whenever a risk is detected, the corresponding lane is tagged "risky" and the ego vehicle should change lane to avoid driving on the risky lane.

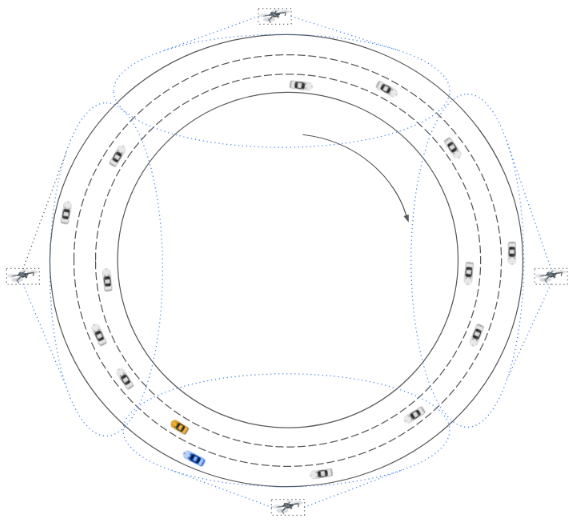

Fig. 4. Simulation scenario: a 4km three-lane circular highway with drones hovering over it

The DQN lane change agent is trained during the simulation. In each simulation step, the agent predicts the next action for the ego vehicle using the DQN and current state retrieved by the Traffic Control Interface (TraCI interface) of SUMO. Then, the ego vehicle updates its state and computes the reward according to (6). The tuple (action, state, reward) is stored in the replay buffer. The details of the performance analysis are provided in the following section.

B. Performance of DEAR

1) Baseline model: We trained several baseline models to compare with our proposed DEAR model as explained below.

- K-Nearest Neighbors (KNN): KNN is a non-parametric supervised machine learning model that uses proximity to classify or predict grouping of individual data points.
- Support Vector Machine (SVM): SVMs are supervised learning models that analyze data for classification and regression problems. They can efficiently perform a non-linear classification by implicitly mapping the inputs into high-dimensional feature spaces.
- Policy Gradient: Policy Gradient is a reinforcement learning algorithm. It increases or decreases the probability of taking an action based on the reward obtained. The reward function is described in (6).
- DEAR-NDR: It is a version of DEAR that does not incorporate drone's assistance nor the density reward R_{den}.

It is to be noted that the KNN and SVM models are built using the scikit-learn package in Python[16]. Moreover, the training set for the baseline models is generated from SUMO with the same setup as DEAR.

2) Performance analysis: In order to access the performance of our proposed lane change agent DEAR, we computed several performance parameters, namely: collision number, average speed, number of lane change requests (LC Request), time spent in risky lanes (Risky Time, t_r) and time spent in front of an emergency vehicle and thus impairing its passage (Blocking Time, t_b) during simulation. Table I, II and III show the performance of the 5 models tested with sparse (50 vehicles), medium (150 vehicles) and dense (250 vehicles) traffic density. The values are averaged over several test episodes. Detailed analysis will be provided in the following paragraphs.

a) Collision number: The main purpose behind our research is to reduce people fatalities related to lane change car accidents, which makes this performance parameter the most important one. According to Tables I, II and III, the DEAR, DEAR-NDR, KNN and SVM models succeed in avoiding collision in sparse, medium and dense traffic scenarios. Conversely, the policy gradient model has an average collision number of 2.3, 6.8 and 6 for the three tested scenarios. It is noteworthy that a collision-less agent is solely important when lane changes occur. One can see that SVM induces zero lane change and thus zero collision. Therefore, we found it relevant to compute LC request number.

b) Number of lane change requests: The number of LC Request reveals the different behaviors of reinforcement and machine learning agents. From the three tables, it can be noticed that SVM has no LC request, while KNN keeps an extremely low number of LC requests, not exceeding 4 in all the three scenarios. This means that SVM and KNN have a tendency towards not allowing lane changes even with risky lane and emergency vehicle behind. On the contrary, DEAR has an average number of LC request of 737.7, 830.3 and 632.8, which is higher than that of KNN and SVM. Indeed, DEAR reinforcement learning agent tries to learn safe and efficient lane change by interacting with the risky lanes, ambulances and surrounding vehicles; which leads the agent to take actions. The similar behavior also applies to Policy Gradient and DEAR-NDR.

c) Average speed: One of the reasons for lane change is overtaking a slow vehicle in front so as to obtain higher speed and reduce the total travel time. This leads us to compute the average speed of the ego vehicle during each test episode. Fig. 5 illustrates that DEAR achieves the highest average speed, followed by SVM. In fact, with DEAR, when facing a risky lane or an emergency vehicle, a negative penalty will be affected to the safety reward; inducing the agent to increase the efficiency reward, and more particularly, the speed reward.

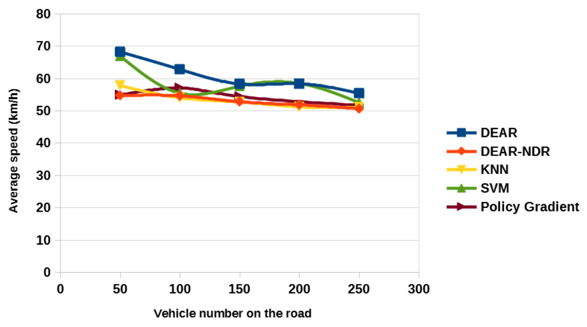

Fig. 5. Average speed of the DEAR, DEAR-NDR, KNN, SVM and Policy Gradient in different traffic density

d) Risky Time: Since one of the added values of our research resides in the agent behavior in a risky environment, we introduce Risky Time. As a matter of fact, t_r indicates how sensitive the agent is to the road risk. According to Tables I and II, DEAR has a relatively low risky time compared to the other models in both sparse and medium traffic. Indeed, our model takes into account the risky time in the reward function. Consequently, the agent will adapt its behavior in the presence of a risky lane and thus pushes the driver to change the lane in the vicinity of an accident or construction works. On the other hand, as presented in Table. III, DEAR drives a comparatively long time on the risk lane than the others in the dense traffic scenario. The reason is that, with the density reward R_{den}, the total safety reward is decreased due to the larger number of vehicle on the road. As a result, the agent has to increase the reward from the efficiency and comfort perspectives, such as achieving higher speed. This leads the agent to keep driving on the leftmost lane, which increases the possibility of driving in the risky lane. Even in this case, the agent still gives way to the emergency vehicles, so we observe a large number of LC Request and a small t_b compared with other models.

e) Blocking Time: The time spent in front of an emergency vehicle is correlated with the driver's cooperation willingness. The higher the cooperation willingness is, the sooner it will change lane to give way. As pointed out by Tables I, II and III, DEAR outperforms other models. Indeed, our agent takes into account the blocking time in the reward function. Consequently, the agent will adapt its behavior in the presence of ambulances and thus gives way to the emergency vehicles regardless of the traffic density. This can be proved by the lowest Blocking Time as compared to the other models that do not take into account the emergency vehicles in the reward function.

TABLE I
MODELS PERFORMANCE TESTED WITH SPARSE TRAFFIC

Model	DEAR	DEAR-NDR	Policy Gradient	KNN	SVM
Collision Number	0	0	2.3	0	0
LC Request	737.7	212.6	342.5	3.6	0
Avg Speed (km/h)	68.2	54.7	54.9	57.8	67
t_r (s)	38.4	47.1	51.9	69.8	66.9
t_b (s)	93.5	173.4	138.3	284.1	123

TABLE II
MODELS PERFORMANCE TESTED WITH MEDIUM TRAFFIC

Model	DEAR	DEAR-NDR	Policy Gradient	KNN	SVM
Collision Number	0	0	6.8	0	0
LC Request	830.3	75.7	316	2.4	0
Avg Speed (km/h)	58.3	52.8	54.4	52.7	57.6
t_r (s)	38.9	46.1	82.2	61.1	42.6
t_b (s)	178.7	291.5	271.4	260.5	249.5

TABLE III
MODELS PERFORMANCE TESTED WITH DENSE TRAFFIC

Model	DEAR	DEAR-NDR	Policy Gradient	KNN	SVM
Collision Number	0	0	6	0	0
LC Request	632.8	61.3	105.3	3.3	0
Avg Speed (km/h)	55.4	50.6	51.7	51.1	52.5
t_r (s)	121.5	34.5	11.8	44.7	78.1
t_b (s)	186.8	366.8	406.9	392.1	556.5

At this stage, we may draw the following conclusions:

- The previous analysis sheds the light on the enhanced performance provided by the drones-assisted platform. Indeed, the frequent computation of the vehicular density by drones highly impacts the reward function. More specifically, a negative penalty is assigned with the sparse, medium and dense traffic; preventing collision and reducing sojourn time on highway. This fact is highlighted by the higher performance of DEAR compared to DEAR-NDR and other models. DEAR succeeds to avoid collision while taking into account the safety, comfort and efficiency perspectives.
- Compared to DEAR-NDR, the dynamic fine-tuning of the collision reward at the drones achieves satisfying performance with DEAR. Indeed, our DEAR agent takes into consideration the fluctuating environment and the collision occurrences at the end of each epoch time. This will adapt the agent to the highway scenario.
- Whether in sparse, medium or dense vehicular traffic scenarios, the proposed DEAR agent is able to learn and

adapt to the complex environment where there are potential risks, urgent lane change demands, and emergency vehicles.

V. Conclusion and Future Work

In this paper, we bring to the focus an innovative LCA platform assisted by drones that collect and process vehicular data, more particularly vehicular density, owing to their Line-of-Sight links and high mobility. This platform is based on a DQN algorithm that efficiently assists the driver to take the decision of changing the lane at the optimal instant of time. This is achieved by an efficient reward function that includes safety, efficiency and comfort. Performance analysis proves that the proposed lane change agent reduces the total travel time while achieving collision-less trips.

To address the real-time stringent requirements needed to lane change decisions, we propose in a second step to tackle federated machine learning: drones will play the role of a central server that aggregates local parameters and trains local models distributed on vehicles. Meanwhile, we are going to adopt the authentic highway dataset, NGSIM, for a more realistic scenario.

References

[1] J. Bie, M. Roelofsen, L. Jin, and B. van Arem, ”Lane Change and Overtaking Collisions: Causes and Avoidance Techniques,” pp 143-187, In: Naja R. (eds) Wireless Vehicular Networks for Car Collision Avoidance. Springer, New York, NY,2013, ISBN:978-1-4419-9562-9.

[2] P. Kumar, M. Perrollaz, S. Lefèvre, and C. Laugier, ”Learning-based approach for online lane change intention prediction,” 2013 IEEE Intelligent Vehicles Symposium (IV), 2013, pp. 797-802, doi: 10.1109/IVS.2013.6629564.

[3] M. Vishal, K. Christos, and A. Constantinos, ”Prediction of Lane-Changing Maneuvers with Automatic Labeling and Deep Learning,” Transportation Research Record 2020, Vol. 2674(7), pp. 336–347, doi: 10.1177/0361198120922210.

[4] A. HongIl and J. Jung, ”Decision-making system for lane change using deep reinforcement learning in connected and automated driving,” Electronics 8.5 (2019): 543.

[5] T. Shi, P. Wang, X. Cheng, C. -Y. Chan, and D. Huang, ”Driving Decision and Control for Automated Lane Change Behavior based on Deep Reinforcement Learning,” 2019 IEEE Intelligent Transportation Systems Conference (ITSC), 2019, pp. 2895-2900, doi: 10.1109/ITSC.2019.8917392.

[6] P. Wang, C. Chan, and A. de La Fortelle, ”A Reinforcement Learning Based Approach for Automated Lane Change Maneuvers,” 2018 IEEE Intelligent Vehicles Symposium (IV), 2018, pp. 1379-1384, doi: 10.1109/IVS.2018.8500556.

[7] C. Hoel, K. Wolff, and L. Laine, ”Automated Speed and Lane Change Decision Making using Deep Reinforcement Learning,” 2018 21st International Conference on Intelligent Transportation Systems (ITSC), 2018, pp. 2148-2155, doi: 10.1109/ITSC.2018.8569568.

[8] J. Wang, Q. Zhang, D. Zhao, and Y. Chen, ”Lane Change Decision-making through Deep Reinforcement Learning with Rule-based Constraints,” 2019 International Joint Conference on Neural Networks (IJCNN), 2019, pp. 1-6, doi: 10.1109/IJCNN.2019.8852110.

[9] Y. Chen, C. Dong, P. Palanisamy, P. Mudalige, K. Muelling, and J. M. Dolan, ”Attention-based Hierarchical Deep Reinforcement Learning for Lane Change Behaviors in Autonomous Driving,” 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2019, pp. 3697-3703, doi: 10.1109/IROS40897.2019.8968565.

[10] W. Guan, H. Jianming, L. Zhiheng, and L. Li, ”Cooperative Lane Changing via Deep Reinforcement Learning,” 2019, doi: https://arxiv.org/pdf/1906.08662

[11] W. Shi, H. Zhou, J. Li, W. Xu, N. Zhang, and X. Shen, ”Drone Assisted Vehicular Networks: Architecture, Challenges and Opportunities,” in IEEE Network, vol. 32, no. 3, pp. 130-137, May/June 2018, doi: 10.1109/MNET.2017.1700206.

[12] M. Volodymyr et al., ”Human-level control through deep reinforcement learning,” Nature 518 (2015): 529-533.

[13] H. Hasselt, A. Guez, and D. Silver, ”Deep reinforcement learning with double Q-Learning,” in Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI’16). AAAI Press, 2094–2100.

[14] F. Ye, X. Cheng, P. Wang, C. -Y. Chan, and J. Zhang, ”Automated Lane Change Strategy using Proximal Policy Optimization-based Deep Reinforcement Learning,” 2020 IEEE Intelligent Vehicles Symposium (IV), 2020, pp. 1746-1752, doi: 10.1109/IV47402.2020.9304668.

[15] P. A. Lopez et al., ”Microscopic Traffic Simulation using SUMO,” 2018 21st International Conference on Intelligent Transportation Systems (ITSC), 2018, pp. 2575-2582, doi: 10.1109/ITSC.2018.8569938.

[16] P. Fabian et al., ”Scikit-learn: Machine learning in Python,” the Journal of machine Learning research 12 (2011): 2825-2830.