TRR
JOURNAL OF THE TRANSPORTATION RESEARCH BOARD

Research Article

Transportation Research Record
2023, Vol. 2677(2) 712–724

DOI: 10.1177/03611981221108377
journals.sagepub.com/home/trr
SAGE

Deep Reinforcement Learning Approach for Automated Vehicle Mandatory Lane Changing

Rami Ammourah[1] and Alireza Talebpour[1]

Abstract

This paper proposes a reinforcement learning-based framework for mandatory lane changing of automated vehicles in a non-cooperative environment. The objective is to create a reinforcement learning (RL) agent that is able to perform lane-changing maneuvers successfully and efficiently and with minimal impact on traffic flow in the target lane. For this purpose, this study utilizes the double deep Q-learning algorithm structure, which takes relevant traffic states as input and outputs the optimal actions (policy) for the automated vehicle. We put forward a realistic approach for dealing with this problem where, for instance, actions selected by the automated vehicle include steering angles and acceleration/deceleration values. We show that the RL agent is able to learn optimal policies for the different scenarios it encounters and performs the lane-changing task safely and efficiently. This work illustrates the potential of RL as a flexible framework for developing superior and more comprehensive lane-changing models that take into consideration multiple aspects of the road environment and seek to improve traffic flow as a whole.

Keywords

operations, automated/autonomous/connected vehicles, automated/autonomous vehicles

Mandatory lane-changing (MLC) maneuvers, on any type of road segment, are considered a challenging task and have been identified as one of the primary sources of shockwave formation and congestion (*1*). Executing safe and efficient MLC maneuvers are even more challenging for connected automated vehicles (CAVs), because of the complex underlying decision-making logic, simultaneous execution of multiple actions, and safety requirements that CAVs have to follow. Such a complicated process can be attributed to the dynamic nature and complexity of road environments and traffic operations (*2*). For instance, vehicles move at different speeds, perform various maneuvers constantly, and the geometry of the road changes continuously. In addition, drivers may behave in a non-cooperative manner and seek to maximize self-benefit rather than behaving in a cooperative, collectively efficient manner.

Several lane-changing models have been proposed in the literature. While the majority of these models are developed for human drivers, many of them have been modified to model the lane-changing behavior of CAVs in various simulation platforms. Gap acceptance-based models are among the most common approaches. In classical gap acceptance models, vehicles make the decision on lane-changing maneuvers based on a critical gap threshold, above which the vehicle would make the lane change and would choose not to otherwise. Ben-Akiva and his colleagues presented several gap acceptance models (e.g., Ahmed et al. [*3*], Mahmassani and Sheffi [*4*], and Ramanujam [*5*]) by introducing an integrated framework that offers a trade-off between mandatory and discretionary lane-changing considerations (*6*). Another study by Abhishek et al. (*7*) aimed to replicate heterogeneous traffic conditions by incorporating constant and variable gap models as well as consistent and inconsistent driver behavior into a single model. Despite widespread adoption, gap acceptance models suffer from a

[1]University of Illinois at Urbana-Champaign, Urbana, IL

Corresponding Author:
Alireza Talebpour, ataleb@illinois.edu

major drawback, that is, most of these models fail to capture the impact of other drivers' behavior (mainly drivers in the target lane) on the lane-changing maneuver (and associated decision-making processes to initiate a lane-changing maneuver).

To address the aforementioned shortcoming, several models have been proposed in the literature that include some measures of driver behavior (before and after the lane-changing maneuvers) and the risk associated with such maneuvers in the modeling process. A well-known example of such models is MOBIL (*8*). MOBIL builds on previous models by incorporating both the utility of a given lane as well as the risk associated with completing a lane-changing maneuver, which is determined by longitudinal accelerations calculated with microscopic traffic models. Another example of such models is the lane-changing model of Talebpour et al. (*9*). They introduced a game-theory-based lane-changing model that considers the impacts of the lane-changing maneuver on the lane-changing vehicles as well as the vehicles directly affected by the maneuver in the target lane (i.e., new follower). They showed that such a framework can significantly improve the accuracy of modeling lane-changing decisions compared with gap acceptance models. Talebpour et al.'s model was later expanded by several other studies, including a study by Kang and Rakha (*10*). They utilized a repeated game framework to model the evolution of drivers' decision-making before and during the lane-changing maneuver. In another recent development, several studies introduced various probabilistic approaches to modeling lane-changing behavior. For instance, Pang et al. (*11*) presented a probabilistic lane-changing model that takes into account past trajectory data in making the probabilistic lane-changing decision. In a similar approach, Park et al. (*12*) built a logistic regression model for lane-changing behavior, where the probability distribution is based on the joint distribution of two main variables, that is, the speed difference and the density difference.

In addition to the aforementioned models, several studies utilized the additional information available through CAVs and the connected driving environment to develop more robust lane-changing models for CAVs. Jin et al. (*13*) proposed a real-time optimal lane selection algorithm by using the information available from connected vehicles. Zheng et al. (*14*) also proposed a cooperative lane-changing strategy in a connected and automated vehicles environment. The strategy was implemented by the coordination of behaviors between merging vehicles and the cooperative vehicle on the target lane. An and Talebpour (*15*) introduced a coordinate merge algorithm based on model predictive control that utilized vehicle-to-vehicle communications to identify the optimal lane-changing trajectory and minimize the impacts of the maneuver on the target lane. Kuefler et al. (*16*) employed generative adversarial networks (GANs) to predict and simulate human driving behavior, including lane-changing maneuvers. A data-driven model based on deep learning was proposed by Xie et al. (*17*) that employs deep belief networks (DBNs) and long short-term memory (LSTM) networks to model the lane-changing process. Ren et al. (*18*) utilized *k*-means clustering to classify driving style before feeding the classified data to a neural network model. Dong et al. (*19*) applied randomized forest and back-propagation neural network (BPNN) algorithms to obtain lane-changing characteristics and apply them to vehicles equipped with cooperative adaptive cruise control (CACC) to improve the efficiency and safety of the lane-changing maneuver.

Moreover, several studies (e.g., Mukadam et al. [*20*], Zhang et al. [*21*], and Wang et al. [*22, 23*]) explored the use of reinforcement learning (RL) to model lane-changing maneuvers. While these RL models share certain similarities, the way environments, states, and actions are defined may vary from one study to another. For instance, the actions chosen by an agent (i.e., a RL vehicle in our domain) may be discrete (move up, down, right, left) or they may be continuous (e.g., choosing a steering angle and an acceleration/deceleration value). For instance, Ye et al. (*24*) designed the action space in both lateral and longitudinal directions. Similarly, reward can be defined in a multitude of ways to achieve the single or multiple tasks available for a given environment. An example of such approaches is the study by Wang et al. (*2*). They created a three-part reward system, which takes into consideration the merge success, merge safety, and merge efficiency.

The majority of existing lane-changing models for CAVs face certain limitations. (1) Most of these models treat lane changing as a binary decision without modeling the lane-changing trajectory and its impacts on the traffic. (2) The limited number of models that generate a lane-changing trajectory do not consider the impact of the lane-changing trajectory on the entire traffic stream in their trajectory generation algorithm. Such a consideration is essential for robust coordinate merge maneuvers. (3) Lane-changing behavior may vary from one instance to another to accommodate environment-specific requirements. The majority of the models do not offer the necessary flexibility to endogenously account for such changes in the lane-changing behavior. Therefore, there is a critical need to develop a generalized flexible lane-changing model that can generate safe trajectories, while accounting for environment-specific challenges and the impact of the trajectory on the entire traffic stream. RL offers a flexible framework to account for various environment-specific needs and to consider the entire traffic stream as part of the reward system. Unfortunately, existing

RL-based lane-changing models fail to provide a realistic representation of this maneuver. Most of these models either define the state space in a discrete manner (e.g., grid space for the coordinates [*23*]) or define the action space in an oversimplified way (e.g., "change lane" or "stay in current lane" [*22*]).

Accordingly, this paper presents a flexible RL-based lane-changing framework, addressing the shortcomings of previous studies. The proposed framework utilizes a continuous state space environment, where vehicle locations are defined by their actual x–y coordinates. The speed and heading of the automated vehicle attempting the merge as well as the location of the immediate leader and follower in the target lane are also included in the state space as continuous values. In addition, the action space is defined as pairs of acceleration/deceleration values and steering angles. While the action space is still discrete, it offers a more realistic representation of vehicle movements compared with existing studies. Note that this additional realism comes at a huge computational cost, since describing the movements of the CAV with a simple "change lane" or "stay in the current lane" significantly reduces the size of action space at each location. Finally, It is important to mention that this study only aims to present a framework and illustrate its capabilities, rather than presenting a ready-to-apply lane-changing model. The remainder of this paper is organized as follows: the next section presents the model formulation and details of the proposed RL-based model. This section is followed by an introduction to the simulation setup, including the RL model parameters. The simulation results and a detailed discussion on the findings of this paper is presented next. Finally, the paper is concluded with summary remarks and future research needs.

Model Formulation

Double Deep Q Network

The double deep Q network (DDQN) (*25*) is an advancement on the original deep Q network (DQN) algorithm (*26*). The DQN combines Q-learning with a deep neural network to perform predictions and make decisions. A DQN agent can learn successful policies directly from high-dimensional sensory inputs using end-to-end RL (*26*). In the DQN algorithm, two neural networks exist: a main network and a target network. The two networks are initialized with random weights, where the input to the networks are the states of the environment and the outputs are the set of actions that can be chosen by the agent. The main network weights are updated according to the Bellman equation (*27*) and the Q values associated with the actions. Figure 1 shows a simple illustration of a DQN. The target network is identical in architecture to the main network but is updated less frequently than the main network (i.e., every N steps, the weights of the main network are copied to the target network). This is done to improve the stability of the learning process and helps the algorithm converge faster by learning more efficiently. The DQN also utilizes a tool called "experience replay" to improve performance. In experience replay, the agent's experiences at each time step are stored in a replay memory, which we then sample from randomly for the Q-learning process instead of just using the current state/action pairs that occur during simulation. On the other hand, the DDQN was proposed by Van Hasselt et al. (*25*) to address some over-estimations that occur in the original DQN algorithm, while also improving its performance. More details on the structure of the DDQN can be found in Van Hasselt et al. (*25*).

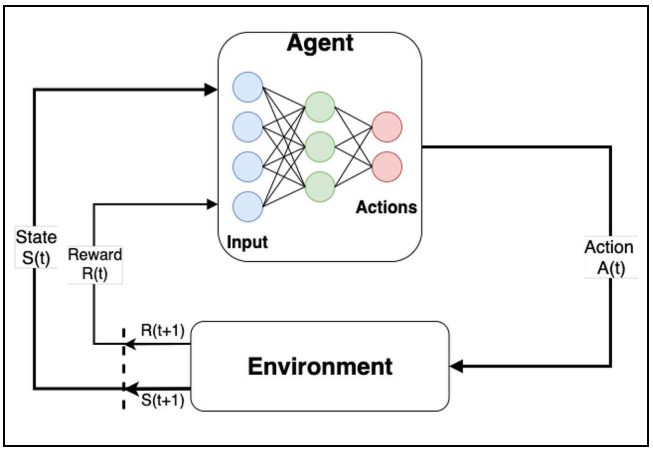

Figure 1. Simple illustration of the deep Q network (*27*).

Model Parameters

The network architecture utilized in this work is a simple fully connected deep neural network. The details of the network architecture are shown in Figure 2. The proposed network was sufficient to achieve favorable results for the task at hand and was chosen over more complex architectures such as convolutional neural networks because of its computational efficiency. Note that a considerable time has been spent on identifying a suitable network structure for the lane-changing problem. Table 1 lists the hyperparameter settings for the formulated DDQN model. We utilize a discount factor, γ, of 0.999, ensuring that the RL agent would strongly consider future rewards when making a decision ($\gamma = 1.0$ means that the agent considers no difference between the current reward and future reward, i.e., the agent becomes more farsighted [*27*]). We do this to give a strong account to the final reward of finishing the task successfully, which we will discuss in detail in later sections. A replay

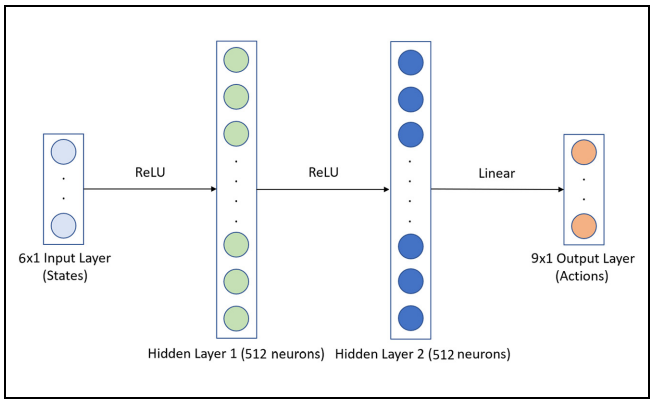

Figure 2. This study's deep Q network architecture.

Table 1. Deep Q Network Hyperparameter Settings

Hyperparameter	Value
Number of layers	2
Number of hidden units	512, 512
Learning rate	0.001
Policy	Epsilon-greedy
ε	1.0 → 0.05
Discount factor γ	0.999
Replay memory size	50,000
Number of episodes	10,000–50,000
Batch size	64
Activation functions	ReLU, linear
Optimizer	Adam

memory size of 100,000 is chosen to stay within computational capacity and is shown to yield good results. Note that larger replay memory size can significantly increase the calibration time and delay the model convergence. As discussed previously, smaller replay memory can result in an undesirable memory loss about effective past actions. Moreover, the agent is set to train for 10 million steps to ensure convergence. The policy selected for learning is the linear-annealed epsilon-greedy policy, where the ε value decreases linearly with the number of steps from 1.0 to 0.05. This ensures the agent explores for an adequate amount of time before starting to follow the greedy action choices, and thus guarantees optimal/near-optimal performance. Note that the minimum ε is set to 0.05 to ensure some level of exploration throughout the calibration process. This is critical to ensure that the system does not stay within a local minimum.

Simulation Setup

Simulation Environment

Our problem is defined within a two-lane environment: a merge lane and a main/target lane. Each lane is 4 m wide, and the merge lane is 200 m long with a 100-m taper section. Figure 3 illustrates the road environment designed for the simulation experiments. No vehicles other than the automated vehicle are present on the merge lane, while other vehicles in the main lane are designed according to the intelligent driver model (IDM) (*28*) to govern their longitudinal motion along the road segment. Main lane vehicles are also modeled to respond to the attempts of the automated vehicle to merge into the target lane. This is done by following a sigmoid cumulative distribution function that controls the probability of a trailing vehicle to switch its leading vehicle from the IDM vehicle ahead (old leader) to the automated vehicle (new leader), depending on how close the automated vehicle is to the main lane.

Reinforcement Learning Environment

The automated vehicle's task is to merge into the target lane safely and efficiently, and continue driving along the target lane until a goal point is reached. The goal point is designed to be 100 m beyond the end of the merge lane. Safety is defined as the ability of the automated vehicle to merge and navigate without crossing the outer borders of the two lanes or colliding with a neighboring vehicle, while efficiency is defined based on traffic state and shockwave formation. A successful episode is achieved if the automated vehicle merges successfully with minimal disturbance, and proceeds to drive safely until reaching the set goal point.

To perform the aforementioned tasks successfully, several components need to be defined appropriately. To begin with, a proper definition of state and action spaces is required in order for the RL agent to be able to learn important features and corresponding best actions. In our study, the state space consists of the x–y coordinates of the RL vehicle, speed and heading of the RL vehicle, and the locations of the leading and trailing vehicles (with respect to the automated vehicle). On the other hand, we define the action space in a more complex manner (compared with existing studies); the action space in our environment is pairs of acceleration/deceleration values and steering angles. The acceleration/deceleration values range among -1, 0 and 1 m/s^2, while the steering angles are either -30, 0, or 30 degrees. The RL agent is responsible for choosing the acceleration/deceleration value as well as the steering angle for the vehicle at each time step (t), which is 0.1 s in this study. This makes the task of lane changing a continuous one, as opposed to simpler definitions that perform the lane changing as a one-step "turn right" or "turn left" command. In addition, we need to create a meaningful reward system that guides the RL agent into eventually performing the task successfully. Thus, a multi-part reward system is

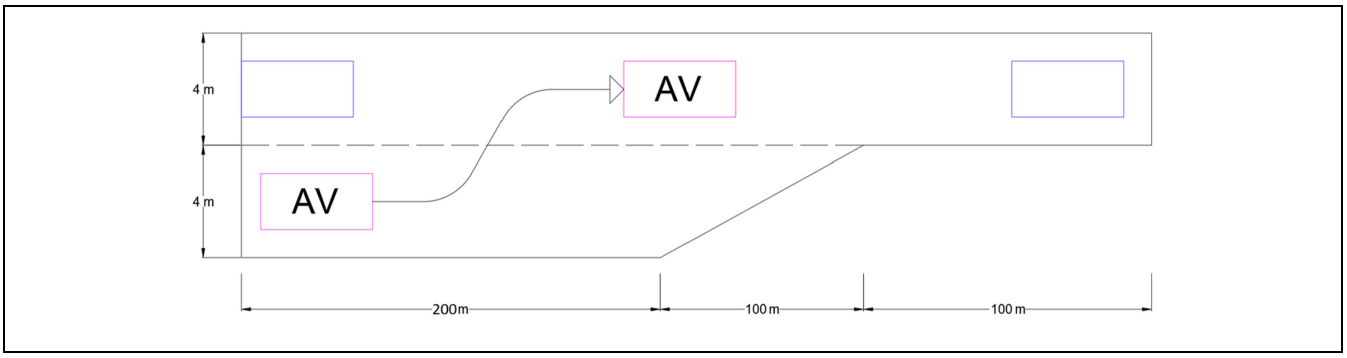

Figure 3. Road environment.
Note: AV = Automated Vehicle, i.e., our ego vehicle.

proposed to tackle this problem. The three main elements of this reward system are a lane-cross negative reward that the RL vehicle incurs whenever it deviates beyond the boundaries of the two lanes, a similar negative reward that is given for any collision that occurs during the merging procedure, and a sizeable positive reward that is awarded at task completion. In addition to the main reward elements, small continuous rewards (which occur every time step) are defined to account for additional requirements concerning efficiency. Those incremental rewards include a small negative reward for every time step the RL vehicle does not finish the task, positive rewards that are added every time step whenever the RL vehicle is within the borders of the target lane and is centered in the target lane, and finally a negative reward that the RL vehicle accrues if it deviates above or below desirable speeds. The first three main rewards ensure a safe lane-changing maneuver, and a well-defined task for the RL agent, while the remaining reward elements ensure an efficient and timely completion of the task. Table 2 presents a detailed overview of the RL and road environment parameters.

We note that the different reward values were updated in an iterative manner as we experimented with a range of values. For example, in earlier stages of training, we only started with the three main reward elements, but as we observed the behavior of the RL agent, we added several other elements. For instance, the negative undesirable speed reward was added after we observed that the RL agent was performing the merging task successfully but then proceeded to slow down heavily to avoid collision. Similarly, multiple other reward elements were incorporated. On the other hand, the specific values of each reward element were chosen using trial and error. It cannot be claimed that this is the optimal reward structure; however, this specific combination of reward values worked for our specific problem. Other reward values may result in comparable and potentially better results. In addition, different scenarios may require some tweaking of the reward structure to meet the objectives of those respective scenarios.

Table 2. Reinforcement Learning and Road Environment Settings

Parameter	Value
Number of states	6
Number of actions	9
Lane-cross reward	−3000
Collision reward	−3000
Merge reward	+50/step
Lane centering reward	+50/step
Undesirable speed reward	−200/step
Noncompletion reward	−20/step
Task completion reward	+10,000
Acceleration/deceleration values (m/ s^2)	−1, 0, 1
Steering angle values (degrees)	−30, 0, 30
Time headway (s)	5, 2.5, 1
Desired velocity (m/s)	30
Time step length (s)	0.1

Results and Discussion

We start by running our model in a trivial environment that contains no vehicles on the entire roadway segment except for the automated vehicle. This was done as a baseline run to verify the ability of the RL agent to learn and perform the lane-changing task successfully. Figure 4 shows the path of the RL vehicle on the two-lane road section. It can be seen that the RL agent learns to make the lane change as soon as possible to avoid a negative reward and then proceeds to drive approximately in the middle of the target lane until reaching the goal point. Note that the deviation from the center of the lane is about ± 0.5 m. Random lateral oscillations in the vehicle's movement can also be seen, which is a product of the inherent randomness of RL (minimum ε is set to

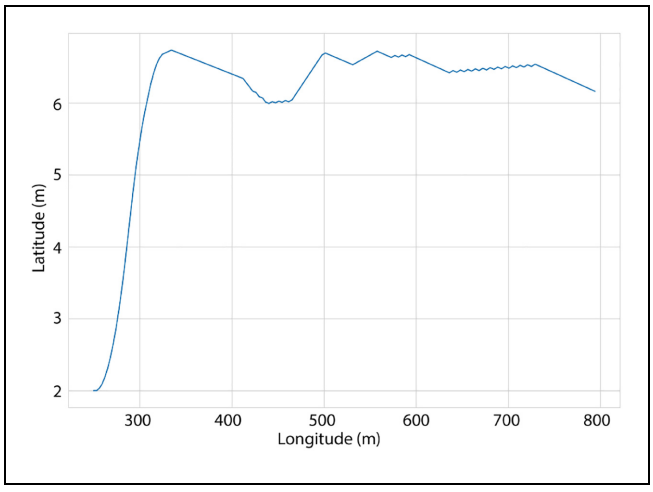

Figure 4. Reinforcement learning vehicle path—empty environment.

0.05, thus at least 5% of vehicle actions are random throughout the training process).

After verifying the ability of the RL agent to learn through the trivial case, we explore three main cases with IDM vehicles in the main lane: (1) 5-s time headway; (2) 2.5-s time headway; and (3) 1-s time headway. To make our problem more challenging, and to test the scalability of the model proposed, we change the initialization location of the RL vehicle in each scenario. To that end, the RL vehicle is initialized at 450 m in the 5-s headway scenario, and at 225 and 90 m for the 2.5- and 1-s scenarios, respectively. The general setup and length of segments remain unchanged. Starting with the 5 s time headway simulation, we only needed to train the model for approximately 10,000 episodes in order for the model to converge and learn optimal policy for this problem. Figure 5*a* presents the reward values during the training process. It can be noted that the curve reaches a near-plateau state toward the end of the training process, while at earlier steps, mostly negative rewards are observed because of high randomness at that phase (exploration phase). Figure 5, *b*–*e*, shows the ability of the RL vehicle to make the lane-changing maneuver without too much effort and without causing disturbances in the target lane. Similar to the empty environment scenario, we can see that the RL vehicle does not strictly remain in the middle of the target lane, which is expected since a slight randomness is inherent in the system and in the way the problem is set up. However, the deviation from the center of the lane is about ± 0.5 m. It should also be noted that scale variance in Figure 5*e* makes the lateral movements of the automated vehicle after making the lane change seem unrealistic. It is noted that slight smoothing was applied to the speed profiles of the RL vehicle in the three scenarios to account for the variability in acceleration/deceleration choices taken by the RL agent, which cause the speed profiles to be somewhat non-smooth and would possibly result in an uncomfortable experience for the passenger. We argue that such smoothing may be advantageous to apply to the model to improve the ride experience. However, of course, we note that further investigation of the effects of such smoothing on the actual operations needs to be considered.

Subsequently, we train our model for the 2.5-s time headway scenario. It can be seen in Figure 6*a* that the training is converging to near-optimal reward values. However, while the 5-s scenario required around 10,000 episodes to train, it is shown here that more than 35,000 episodes were required. On the other hand, as observed in the rest of the figures, the RL vehicle can be seen to be performing the task of lane changing in this scenario quite well. Specifically, the speed and acceleration profiles in Figure 6, *b* and *c*, illustrates that the RL vehicle had minimal impact on main lane traffic and that even the vehicle directly behind it (i.e., IDM-2) was almost unaffected by the merging maneuver. Note that the abrupt changes in the acceleration of this vehicle are in response to the abrupt changes in RL vehicles' acceleration (because of discrete action–acceleration value–space). We also show here, as a representative example, the steering profile for the RL vehicle in Figure 6*f*. We can see that in the early steps of the task, the RL vehicle mostly chooses a steering angle of $+30$ degrees, which corresponds to turning left and quickly completing the merge maneuver. This is followed by -30 degrees which corresponds to correcting its trajectory to follow the vehicles in the main lane after completing the merge maneuver. Finally, the steering angle profile can be seen to nearly converge to 0 degrees. The steering angle profile was smoothed because the discrete nature of the action space forces the RL agent to jump between -30, 0, $+30$, while a smoothed profile shows the trend of actions taken by the agent. Note that the steering angle profile is consistent with the RL vehicle path shown in Figure 6*e* and is similar in other scenarios.

Finally, we explore the case of 1-s time headway between the vehicles and test the ability of the RL agent to learn the proper actions needed to perform the merging task in this challenging scenario. Figure 7*a* shows the trend of the reward gained during the training process. In this case, more steps are required until the reward per episode starts stabilizing and converging to the optimal value after yielding a negative reward for a significant number of episodes. In order for the RL agent to learn to perform the task successfully within this more challenging scenario, we trained the model for approximately 50,000 episodes, which explains the steadier trend when compared with the previous scenario. Figure 7 also

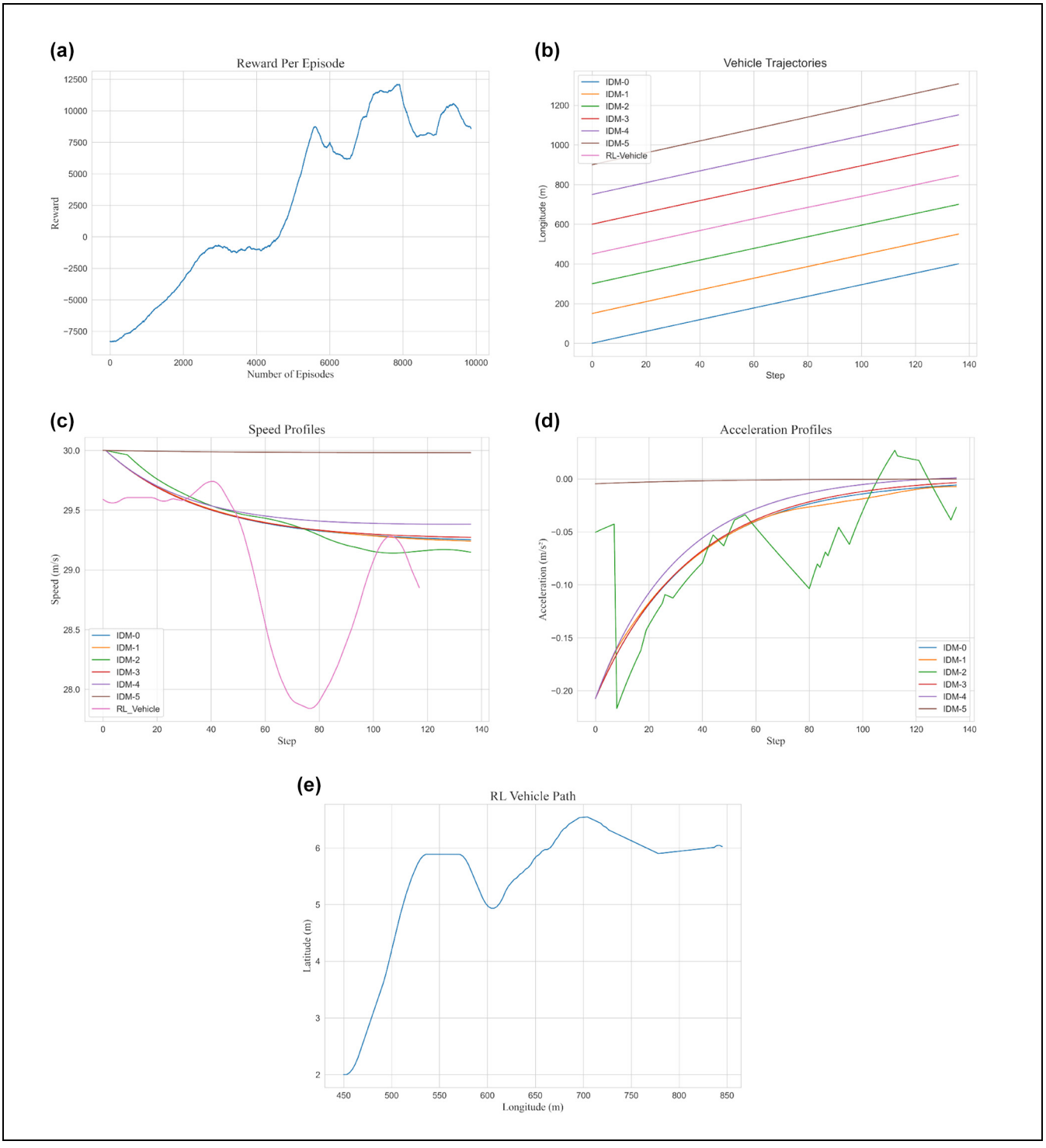

Figure 5. The 5-s headway scenario: (*a*) training reward; (*b*) vehicle trajectories; (*c*) speed profiles*; (*d*) acceleration profiles; (e) reinforcement learning (RL) vehicle path.

Note: IDM = intelligent driver model.

*The speed profile of the RL vehicle appears to fall short because smoothing was applied to it.

shows the vehicle trajectories, speed profiles, acceleration profiles, and the RL vehicle path in the 1-s time headway scenario. It can be observed that because of the small time headway between all the vehicles, the IDM directs all the vehicles (except the leader—IDM-5) to slow down for a portion of time at the beginning of the simulation

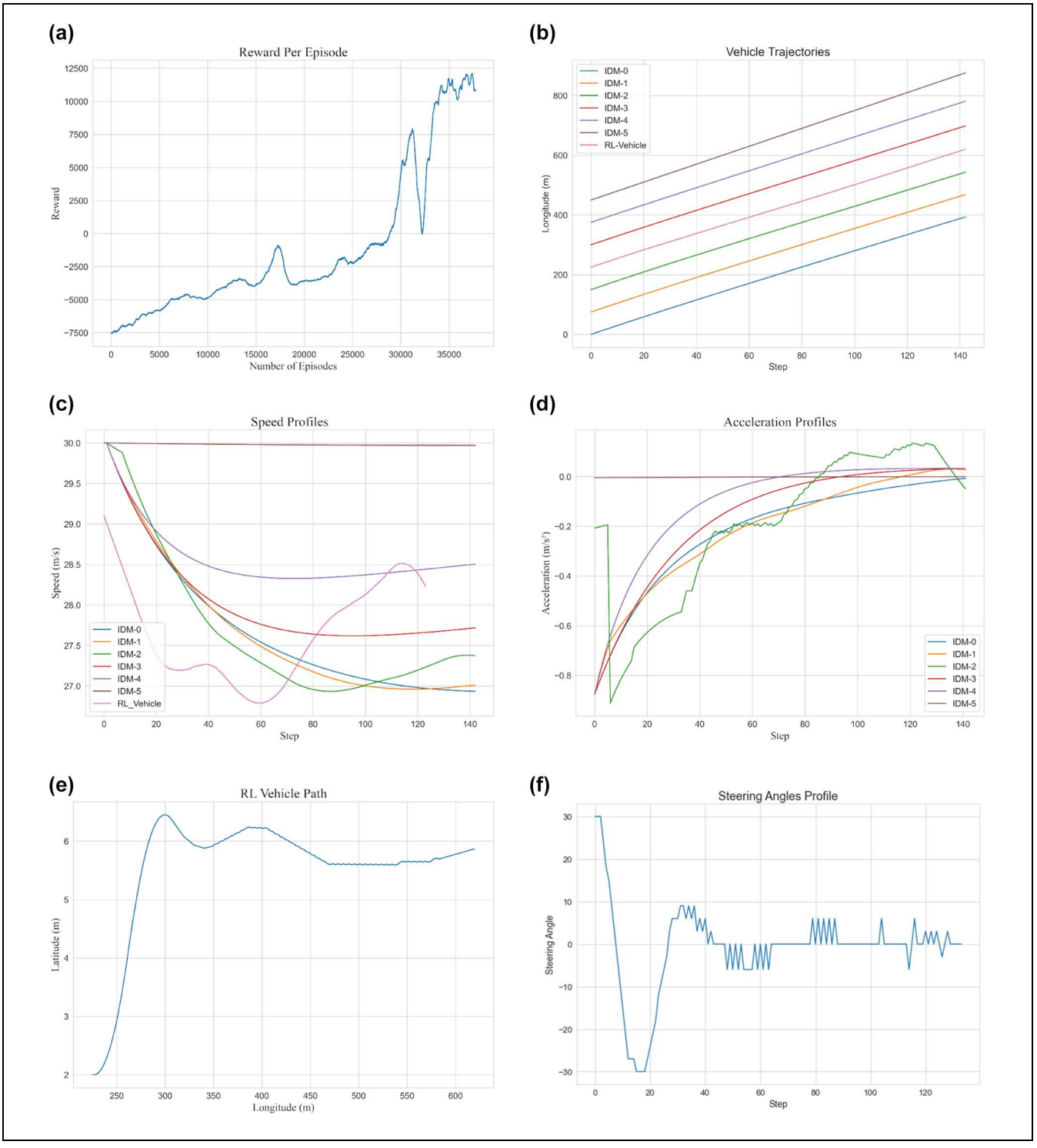

Figure 6. The 2.5-s headway scenario: (*a*) training reward; (*b*) vehicle trajectories; (*c*) speed profiles*; (*d*) acceleration profiles; (*e*) reinforcement learning (RL) vehicle path; (*f*) steering angle profile*.
Note: IDM = intelligent driver model.
*The speed and steering angle profiles of the RL vehicle appear to fall short because smoothing was applied to them.

until an equilibrium headway starts forming between vehicles. This happens because the desirable time headway used for the IDM was set to 1.5 s. In addition, it can be seen that the RL vehicle learns to conduct the lane-changing maneuver with minimal impact on the traffic stream, which can be observed from the speed profiles

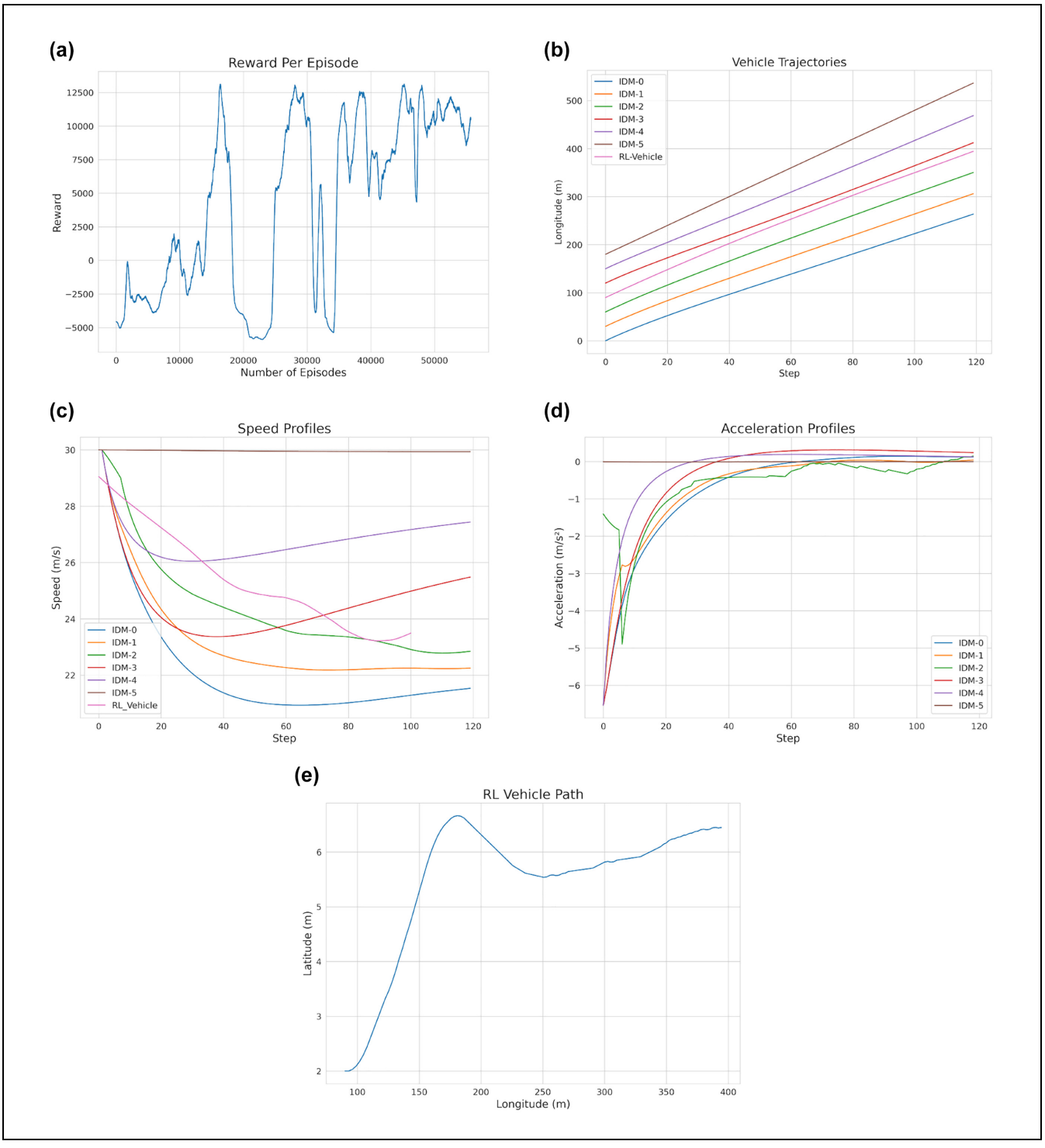

Figure 7. The 1-s headway scenario: (*a*) training reward; (*b*) vehicle trajectories; (*c*) speed profiles*; (*d*) acceleration profiles; (e) reinforcement learning (RL) vehicle path.

Note: IDM = intelligent driver model.

*The speed profile of the RL vehicle appears to fall short because smoothing was applied to it.

(Figure 7*b*). It can also be seen that the RL vehicle decelerates more gradually relative to neighboring vehicles, and maintains a higher speed that results in controlling the shockwave propagation. However, despite a controlled deceleration of the RL vehicle, because of the very small time headway in the target lane, the vehicles directly

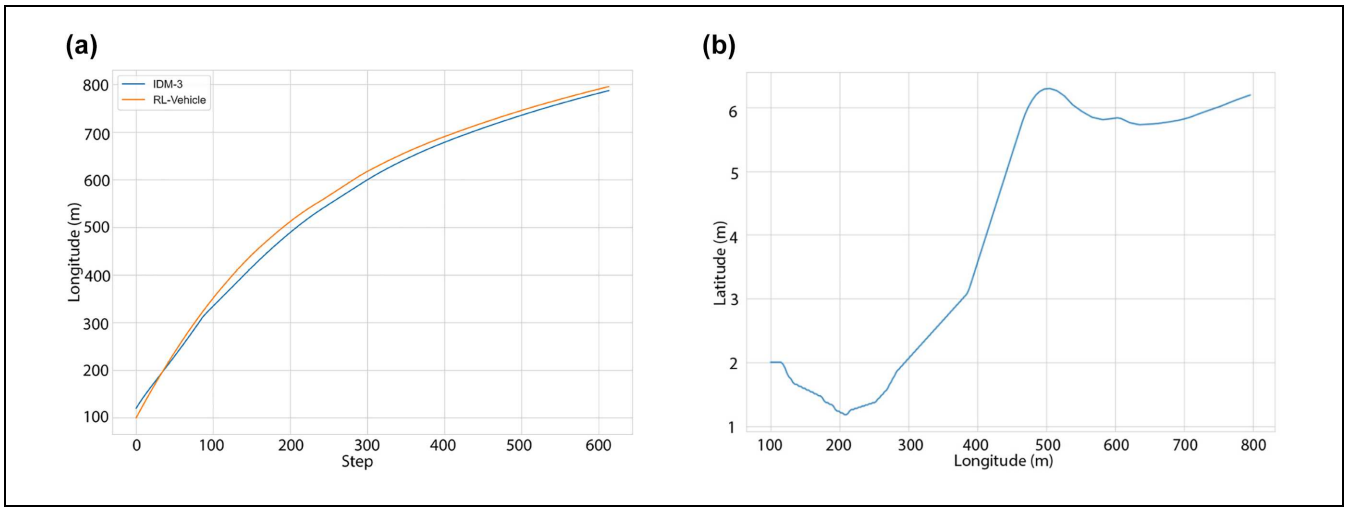

Figure 8. Alternative 1-s headway scenario: (*a*) reinforcement learning (RL) vehicle path; (*b*) vehicle trajectories.
Note: IDM = intelligent driver model.

affected by the lane changing show significant deceleration values for a very short period of time. Note that such a large deceleration can be avoided by replacing IDM vehicles with a more robust platooning algorithm (e.g., the MPC-based model of An and Talebpour [*15*]).

Further simulation runs and exploration of this scenario reveals an interesting behavior that the RL agent occasionally learns to perform on its own. Figure 8*a* shows that because of the small time window the RL vehicle has to complete the lane-changing movement within, it learns to navigate longer through the merge lane before taking the decision to overtake its leading vehicle and making the lane-changing maneuver in a slot ahead of its starting position (with respect to the platoon). The overlap in the trajectories between the RL vehicle and the vehicle ahead (IDM-3) is not a sign of collision. In fact, each vehicle is moving in its own respective lane (only the longitudinal position is plotted here) and it is an overtaking maneuver, which can be verified by Figure 8*b*. This figure shows that the RL vehicle actually remains within the merge lane well beyond the overlap point of approximately 200 m.

While the presented results seem promising and can certainly be improved by fine-tuning the reward function and hyper parameters, the outcome of calibration is not always satisfactory (mainly because of the randomness involved in the action selection mechanism). Accordingly, we also show hereafter that some disadvantages exist with applying a RL approach to model the lane-changing behavior. Figure 9 shows one of the simulation runs for the RL vehicle. It can be seen that after completing a successful lane change, the agent randomly decides to start turning right and almost leaves the target lane before finally swaying back up again and finishing the task. While the RL agent did perform the task successfully and managed to gain a favorable reward for its actions, it is apparent that such behavior is not desirable and has no reason to occur in a realistic setting. Alternatively, while most runs on the 5-s scenario showed no impact of the lane-changing maneuver on the main lane, in some cases, the RL vehicle makes the decision to slow down, which ultimately slows down the whole stream of traffic, even though by looking at the vehicle path alone, one would expect the RL vehicle to be performing better. Figure 10, *a* and *b*, illustrates the aforementioned case.

In addition to the aforementioned challenges, several other issues were faced during the training and testing phases of the RL model. For instance, some runs resulted in falling into local minima that at first glance gave the impression that the model was converging. However, on further examination and visualization of the vehicle path and trajectory, it was found that the RL vehicle was learning a sub-optimal policy and, in reality, was not achieving the task originally described. In addition, substantial computational capacity is required to run the simulation and train the model for a sufficient amount of time. While we sometimes selected the number of training steps to be in excess of 50,000 episodes, earlier stages of training were run with much fewer steps. That resulted in a wider range of randomness and variance in the results, which required increasing the number of steps until a more stable model was produced. In other words, most of the above challenges were solved by providing the right balance between exploring the environment and exploiting the findings from previous experiments.

On the other hand, another critical aspect to deep RL models is training initialization and the randomness in some parameter choices. While most parameters can

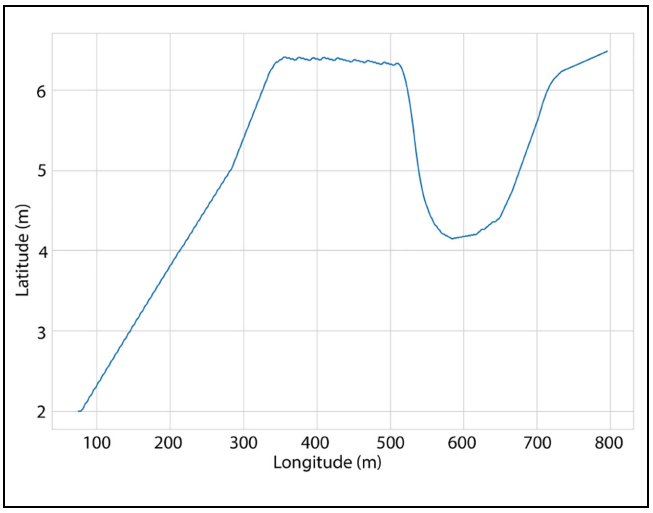

Figure 9. Case 1: unsatisfactory vehicle path.

have a solid mathematical reasoning behind their selection, there can still exist some trial and error. For instance, our final choice of 0.999 for the discount factor, γ, was made after several trials of values ranging between 0.70 and 1.0. Our choice of 0.999 ultimately provided the most favorable results among the different scenarios explored. This may be the case since we explicitly define a "task success" reward, which needs to be accounted for somewhat significantly to guide the agent toward finishing the task successfully. Note that in other scenarios and problem definitions, such an end-of-task reward might not exist, and thus, a lower discount factor might become a better choice. Alternatively, the decay method for ε can also result in different outcomes. We have selected a linearly decaying epsilon; however, other methods such as step decay may have different outcomes, such as faster or slower convergence of the model. This is essential, as we have discussed earlier, where the model could converge to undesirable local minima. It is also worth mentioning that for some scenarios, it may be infeasible (a non-convex setting) or the deep RL model may be unable to find the global optimal solution for the problem. While it may reach near-optimal results, in some real-world situations, some near-optimal solutions may not perform reasonably. This was apparent in many instances of our training and testing procedure, where the RL agent, for example, would perform the task of lane changing successfully but would continue to collide into nearby vehicles. Accordingly, it is critical to capture such instances that may lead to catastrophic outcomes in a real-world setting. Note that the reward setup also plays an important role since, for instance, the RL agent may be achieving a higher reward for completing the lane change while it may not care about the penalty of colliding.

Conclusion and Future Work

We proposed a deep RL-based algorithm utilizing the DDQN for automated vehicle merging in a decentralized non-cooperative manner. The proposed approach utilizes a continuous state space and more realistic action space compared with previous studies. For the RL agent, we defined a six-element state space consisting of the x–y coordinates of the RL vehicle, the speed and heading of the RL vehicle, and the locations of the leading and trailing vehicles on the target lane. We also defined the action space in a more realistic manner than before where the model has to choose between nine different action pairs at each time step. Each action pair consists of a combination of an acceleration/deceleration value (−1, 0, or

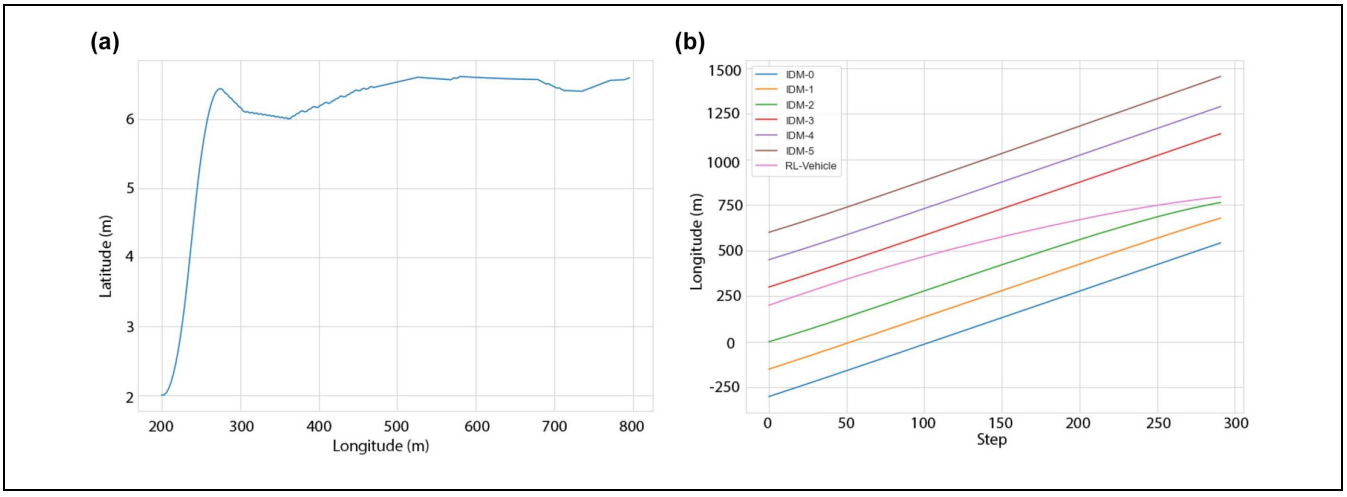

Figure 10. Case 2: unsatisfactory vehicle trajectories: (*a*) reinforcement learning (RL) vehicle path; (*b*) vehicle trajectories.
Note: IDM = intelligent driver model.

$1\,m/s^2$) and a steering angle value (-30, 0, or 30 degrees). Therefore, we take the task of lane changing one step closer to realism by letting the RL agent choose its action every time step (i.e., 0.1 s) and move along the lanes continuously, while monitoring its location and the locations of neighboring vehicles. We demonstrate the model's ability to learn the proper actions it requires to perform the lane-changing maneuver under different circumstances and scenarios. Finally, we propose a reward system that allows the RL agent to perform the task in a timely and efficient manner.

Incorporating additional components to the reward structure to fine-tune the RL vehicle behavior and including the impacts of the RL vehicle on the entire traffic stream as part of the decision-making have been left for future research.

Author Contributions

The authors confirm contribution to the paper as follows: study conception and design: Rami Ammourah, Alireza Talebpour; data collection: Rami Ammourah, Alireza Talebpour; analysis and interpretation of results: Rami Ammourah, Alireza Talebpour; draft manuscript preparation: Rami Ammourah, Alireza Talebpour. All authors reviewed the results and approved the final version of the manuscript.

Declaration of Conflicting Interests

The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The authors disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This material is based on work supported by the National Science Foundation under Grant No. 2047937.

ORCID iDs

Rami Ammourah https://orcid.org/0000-0003-0912-8265
Alireza Talebpour https://orcid.org/0000-0002-5412-5592

References

1. Cambridge Systematics, Inc. *Traffic Congestion and Reliability: Trends and Advanced Strategies for Congestion Mitigation*. FHWA-HOP-05-064. Federal Highway Administration, Office of Operations, Washington, D.C., 20590, 2005.
2. Wang, H., S. Yuan, M. Guo, X. Li, and W. Lan, A Deep Reinforcement Learning-Based Approach for Autonomous Driving in Highway On-Ramp Merge. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, Vol. 235, No. 10–11, 2021, pp. 2726–2739.
3. Ahmed, K., M. Ben-Akiva, H. N. Koutsopoulos, and R. Mishalani. Models of Freeway Lane Changing and Gap Acceptance Behavior. *Transportation and Traffic Theory*, Vol. 13, 1996, pp. 501–515.
4. Mahmassani, H., and Y. Sheffi. Using Gap Sequences to Estimate Gap Acceptance Functions. *Transportation Research Part B: Methodological*, Vol. 15, No. 3, 1981, pp. 143–148.
5. Ramanujam, V. *Lane Changing Models for Arterial Traffic*. Doctoral dissertation. Massachusetts Institute of Technology, Cambridge, 2007.
6. Ben-Akiva, M., C. Choudhury, and T. Toledo. Lane Changing Models. *Proc., International Symposium of Transport Simulation*, Lausanne, Switzerland, 2006.
7. Abhishek, A., M. Boon, and M. Mandjes. Generalized Gap Acceptance Models for Unsignalized Intersections. *Mathematical Methods of Operations Research*, Vol. 89, 2019, pp. 385–409.
8. Kesting, A., M. Treiber, and D. Helbing. General Lane-Changing Model MOBIL for Car-Following Models. *Transportation Research Record: Journal of the Transportation Research Board*, 2007. 1999: 86–94.
9. Talebpour, A., H. S. Mahmassani, and S. H. Hamdar. Modeling Lane-Changing Behavior in a Connected Environment: A Game Theory Approach. *Transportation Research Procedia*, Vol. 7, 2015, pp. 420–440.
10. Kang, K., and H. A. Rakha. Modeling Driver Merging Behavior: A Repeated Game Theoretical Approach. *Transportation Research Record: Journal of the Transportation Research Board*, 2018. 2672: 144–153.
11. Pang, M.-Y., B. Jia, D.-F. Xie, and X.-G. Li. A Probability Lane-Changing Model Considering Memory Effect and Driver Heterogeneity. *Transportmetrica B: Transport Dynamics*, Vol. 8, No. 1, 2020, pp. 72–89.
12. Park, M., K. Jang, J. Lee, and H. Yeo. Logistic Regression Model for Discretionary Lane Changing Under Congested Traffic. *Transportmetrica A: Transport Science*, Vol. 11, 2015, pp. 333–344.
13. Jin, Q., G. Wu, K. Boriboonsomsin, and M. Barth. Improving Traffic Operations Using Real-Time Optimal Lane Selection With Connected Vehicle Technology. *Proc., 2014 IEEE Intelligent Vehicles Symposium*, Dearborn, MI, IEEE, New York, 2014, pp. 70–75.
14. Zheng, Y., B. Ran, X. Qu, J. Zhang, and Y. Lin. Cooperative Lane Changing Strategies to Improve Traffic Operation and Safety Nearby Freeway Off-Ramps in a Connected and Automated Vehicles Environment. *IEEE Transactions on Intelligent Transportation Systems*, Vol. 21, 2019, pp. 4605–4614.
15. An, G., and A. Talebpour. Lane-Changing Trajectory Optimization to Minimize Traffic Flow Disturbance in a Connected Automated Driving Environment. *Proc., IEEE Intelligent Transportation Systems Conference (ITSC)*, Auckland, New Zealand, IEEE, New York, 2019, pp. 1794–1799.
16. Kuefler, A., J. Morton, T. Wheeler, and M. Kochenderfer. Imitating Driver Behavior With Generative Adversarial Networks. *Proc., IEEE Intelligent Vehicles Symposium (IV)*, Los Angeles, CA, IEEE, New York, 2017, pp. 204–211.

17. Xie, D.-F., Z.-Z. Fang, B. Jia, and Z. He. A Data-Driven Lane-Changing Model Based on Deep Learning. *Transportation Research Part C: Emerging Technologies*, Vol. 106, 2019, pp. 41–60.
18. Ren, G., Y. Zhang, H. Liu, K. Zhang, and Y. Hu. A New Lane-Changing Model With Consideration of Driving Style. *International Journal of Intelligent Transportation Systems Research*, Vol. 17, 2019, pp. 181–189.
19. Dong, C., H. Wang, Y. Li, X. Shi, D. Ni, and W. Wang. Application of Machine Learning Algorithms in Lane-Changing Model for Intelligent Vehicles Exiting to Off-Ramp. *Transportmetrica A: Transport Science*, Vol. 17, No. 1, 2021, pp. 124–150.
20. Mukadam, M., A. Cosgun, A. Nakhaei, and K. Fujimura. Tactical Decision Making for Lane Changing With Deep Reinforcement Learning. *Proc., 31st Conference on Neural Information Processing Systems (NIPS 2017)*, Long Beach, CA, 2017.
21. Zhang, S., H. Peng, S. Nageshrao, and E. Tseng, Discretionary Lane Change Decision Making Using Reinforcement Learning With Model-Based Exploration. Proc., 18th *IEEE International Conference on Machine Learning and Applications (ICMLA)*, Boca Raton, FL, IEEE, New York, 2019, pp. 844–850.
22. Wang, J., Q. Zhang, D. Zhao, and Y. Chen. Lane Change Decision-Making Through Deep Reinforcement Learning With Rule-Based Constraints. *Proc., International Joint Conference on Neural Networks (IJCNN)*, Budapest, Hungary, IEEE, New York, 2019, pp. 1–6.
23. Wang, G., J. Hu, Z. Li, and L. Li. Cooperative Lane Changing via Deep Reinforcement Learning. *arXiv Preprint arXiv:1906.08662*, 2019.
24. Ye, F., P. Wang, C. Chan, and J. Zhang. Meta Reinforcement Learning-Based Lane Change Strategy for Autonomous Vehicles. *arXiv Preprint arXiv:2008.12451*, 2020.
25. Van Hasselt, H., A. Guez, and D. Silver. Deep Reinforcement Learning With Double q-Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 30, No. 1 2016. https://ojs.aaai.org/index.php/AAAI/article/view/10295.
26. Mnih, V., K. Kavukcuoglu, D. Silver, A. Rusu, J. Veness, M. Bellemare, A. Graves, et al. Human-Level Control Through Deep Reinforcement Learning. *Nature*, Vol. 518, 2015, pp. 529–533.
27. Sutton, R. S., and A. G. Barto . *Reinforcement Learning: An Introduction*, 2nd ed. MIT press, Cambridge, MA, 2018.
28. Treiber, M., A. Hennecke, and D. Helbing. Congested Traffic States in Empirical Observations and Microscopic Simulations. *Physical Review E*, Vol. 62, 2000, pp. 1805–1824.