Enhancing Reinforcement Learning for Automated Driving through Virtual Lane Logic

Enhancing Reinforcement Learning for Automated Driving through Virtual Lane Logic ⋆

A. Fasiello * **F. Cerrito** * **V. Razza** ** **M. Canale** *

* *Department of Control and Computer Engineering, Politecnico di Torino, Corso Duca degli Abruzzi 24, 10129 Torino, Italy.*
** *Department of Management and Production Engineering, Politecnico di Torino, Corso Duca degli Abruzzi 24, 10129 Torino, Italy.*

Abstract: This work investigates an alternative approach to current control systems for the Automated Driving (AD) of shuttle vehicles on dedicated roads. The proposed solution decouples the problem into two levels: a Deep Deterministic Policy Gradient (DDPG) Reinforcement Learning (RL) agent and a dedicated vehicle logic generating Virtual Lane (VL) data to eliminate redundancy and allow for smooth lane changes on curved roads. The training uses an environment defined through a model-based simulation, exploiting MATLAB Inc. (2020) and Simulink tools, and has been conducted following a Curriculum Learning strategy. The performance of the introduced approach have been evaluated by testing the agent capabilities and exploring its behavior in the presence of external disturbances in the controlled states.

Keywords: Automated Driving, Reinforcement Learning, Deep Deterministic Policy Gradient.

1. INTRODUCTION

Systems for Automated Driving (AD) leverage a great variety of cutting-edge technologies to perform the principal Dynamic Driving Tasks (DDTs) such as Adaptive Cruise Control (ACC), Lane Keeping (LK), and Lane Change (LC), allowing them through a data elaboration pipeline of sensing, perception and localization, scene representation, planning and decision, and, finally, control (see, e.g., Watzenig and Horn (2016)).

The possible applications of Reinforcement Learning (RL) in this area have led to strong interest in recent years. A great variety of different implementations, methodologies, and paradigms can be found in literature: path, trajectory, and motion planning, optimization of actuation control, complex navigation and traffic decision-making, and lane position control are just a few examples outlined by Kiran et al. (2021) survey. The solutions developed are several, mainly divided between exploiting RL algorithms as support for the different control systems, like the deep RL framework for eco-driving proposed by Zhu et al. (2024), and direct RL control of the vehicle through agent policy actions.

One of the most common driving scenarios in RL for AD literature is the racing one, where the need to run with the highest feasible speed while avoiding crashes is easily modeled by a reward function. Contributions in this field, oriented toward the RL control, exploit mainly a black-box approach, where the scenario readings given by the simulator (containing data about track axis distance and orientation, range finder sensors, and vehicle telemetry states) are directly used as observation input by the agent without any pre-elaboration: both Wang et al. (2018), and Hua et al. (2022) present Deep Deterministic Policy Gradient (DDPG) agents for fast driving on the simulated racing circuit (without other competitor cars during the training phase) controlling acceleration, brake, and steering angle of the vehicle. In both cases, the system reward was focused on the amplitude of the longitudinal velocity and the punishment of the lateral deviation from the road axis. Moreover, Hua et al. (2022) proposed an improved exploration strategy through a noise structure based on the Stanley method, modifying the random signal superposed to the action to allow for exploration in a deterministic agent into a correcting contribution, forcing the experiencing of just the useful state-action combinations. This system resulted in faster and more effective learning, showing a learning speed double than the one proposed by Wang et al. (2018).

The highway environment represents one of the AD most important fields of application. Therefore, the literature provides several examples of RL being exploited for direct control or support of different subsystems for DDTs. Wang et al. (2020) proposed an RL system for LK, comparing the performances of Deep Q-Network (DQN) and DDPG algorithms on a simulation system where the vehicle, initialized in a random heading angle and tracking error, is required to follow a highway-like road with constant curvature, controlling the steering angle. In a so simplified problem, DQN resulted in having the fastest training while DDPG resulted in the best performances, achieving a better zero tracking of the lateral error.

The urban environment is the most complex, characterized by sharper curves and complex interactions with other actors on the road. In those scenarios, it is possible to find the exploitation of more structured learning methods. Anzalone et al. (2021) propose a complete black-box approach where a Proximal Policy Optimization (PPO) agent directly receives as input the images given by the camera sensors simulated by the system (along with vectors of road, vehicle, and navigational features). They used *Curriculum Learning* by Bengio et al. (2009), consisting

⋆ This work is partly supported by the project Piano Nazionale di Ripresa e Resilienza (PNRR)-Next Generation Europe, which has funded by the European Union and the Italian Ministry of University and Research – DM 117/2023.

of dividing the complex problem into stages characterized by growing difficulty. The exploitation of images as observation input introduced the need to train the network, not only in increasingly complex situations, but also in all possible weather and light conditions that could affect the data experienced.
This paper presents an alternative RL-based AD approach inspired by optimal control methods(see Canale and Razza (2024)). It aims to mimic their performances while drastically lowering the computational effort required for the online computation of control signals. The approach exploits already available data from state-of-the-art lane boundary and road geometry recognition systems implemented on modern cars. This prevents a black-box solution that, starting from raw data like a camera image, directly computes the vehicle control input. The advantages are two-fold: the algorithm can focus on specific tasks while, on the other hand, the structure is kept simple. As to the cited RL-based solutions, the work aims to present a new framework where a suitable Virtual Lane Logic (VLL) supports the learning of the RL agent and its control of the system, generating data for executing LC tasks smoothly and safely, even in the case of curved roads. This enhancement, while reducing the size of the needed observation input of the agent, avoid the utilization in an urban environment of more complex neural networks such as the convolutional ShuffleNet used by Anzalone et al. (2021) or recurrent ones.

2. PROBLEM SETTING

The goal of this work is to develop an AD solution for shuttle vehicles oriented to the fourth SAE International (2021) level, capable of driving on dedicated roads. In this context, the proposed solution focuses on a low-complexity scenario where the shuttle operates in a restricted area without vulnerable road users. The vehicle can perform a complete range of maneuvers. In particular, it can start and stop at a pre-determined point and merge into and out of the main traffic lane. To meet all the application requirements, restrictions are placed on the vehicle's behavior, such as limiting the maximum speed to 50 km/h. To ensure passenger comfort based on De Winkel et al. (2023), the vehicle's longitudinal acceleration is limited in $[-3, 3]$ m/s^2, and the steering speed is limited to [-8,8] rad/s. Implementing such constraints not only enhances comfort but also speeds up the training phase by minimizing steering oscillation, which can lead to undesired or unstable behavior. The execution of emergency maneuvers, such as emergency braking or evasive maneuvers, which require action beyond the prescribed limits, is outside the scope of this study.
The proposed solution is developed and tested on a properly designed scenario. The shuttle route is placed in Torino, Italy, in a neighborhood of Politecnico area (see Fig. 1), with 12 strategic shuttle stop stations in an urban scenario. This road network is ideal for testing AD solutions thanks to the presence of vehicle-to-everything (V2X) facilities. In a V2X application framework, the infrastructure provides the vehicle with all information about speed limits and destination locations. The infrastructure also provides constraints on the required path length and the target lane for LC maneuvers. For simplicity, the sensing and sensor fusion phases of the pipeline will not be addressed. Direct data reading from

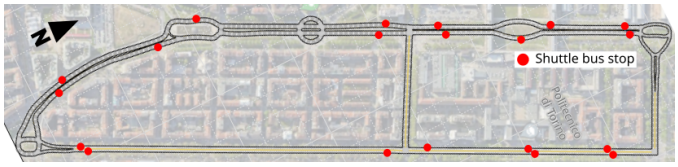

Fig. 1. Bird's-eye view of the target circuit with interest map section of Torino, Italy, in the background.

the scenario is exploited to reduce the computational effort required by the system to simulate each episode of the learning, reducing the time required by the training.
The VLL can be suitably adapted to different scenario conditions and provided data. In the proposed solution, the agent receives the Virtual Lane (VL) data as input, as opposed to raw and redundant information produced by sensors. Thus, the VLL represents an intermediate layer capable of extending the usability of the trained networks to different data structures. The proposed framework separates the problems of planning and control. This increases the flexibility and explainability of the solution, enhancing the ability to understand the controller's behavior and intervene in case of evident biases.
All the training and the simulation are performed in MATLAB and Simulink, exploiting the Automated Driving and the Reinforcement Learning toolboxes. This allows us to maintain efficient monitoring of the data flow of the simulation while providing a user-friendly development tool for the definition of the DDPG agent and the creation of the driving scenario, together with the extraction of road networks from Haklay and Weber (2008) open-source project OpenStreetMap.

3. VLL-RL METHOD

This section describes the proposed control architecture for the considered AD application based on the DDPG agent and the VLL.

3.1 Architecture

As shown in Fig. 2, the vehicle DDTs (e.g., ACC, LK, and LC) are managed by the proposed DDPG agent in coordination with the VLL. The VLL is implemented onboard and receives data from the infrastructure and the vehicle sensors (which measure the lane boundaries). It computes the desired VL path for the RL agent, which provides the vehicle control inputs, i.e., longitudinal acceleration and steering speed.

3.2 Reinforcement Learning Basic Concepts

RL is a machine learning technique where an *agent* is trained to perform a task through trial-and-error interaction with the environment (see, e.g., Kaelbling et al. (1996)). In general, the agent performs an action based on the observed environment, through a dynamical mapping which is adjusted to maximize a return. In this context, the actions are the control output variable, while the environment is everything not under the direct control of the agent itself, including the vehicle dynamics. Throughout this process, the agent continuously balances the need to exploit its current knowledge (exploitation) and the need to gather new information (exploration).

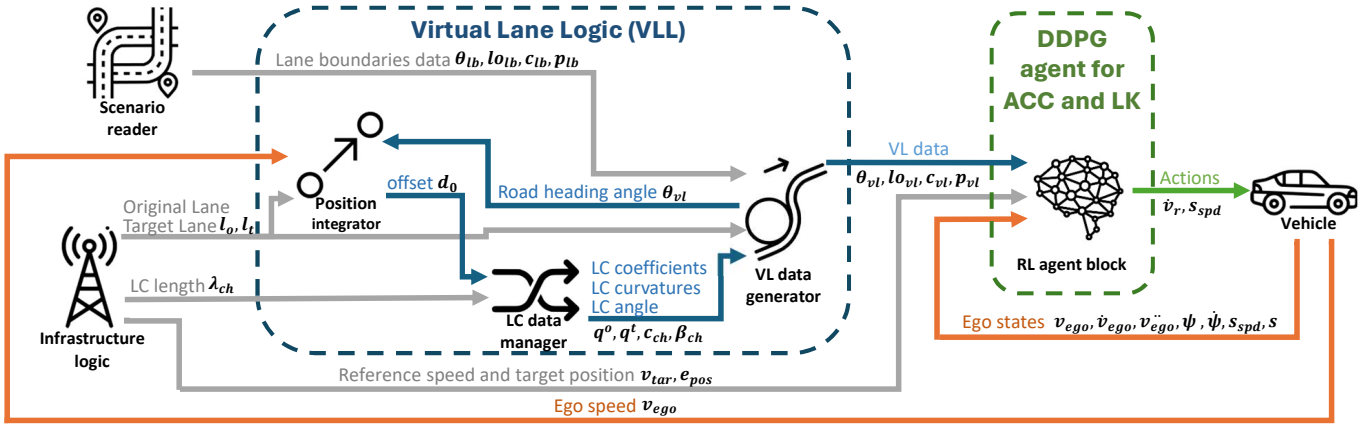

Fig. 2. Main flow of data inside the control architecture of the solution.

A critical aspect of RL-based solution design is selecting an efficient RL algorithm from the numerous available ones. Input and Output signals of the system have been modeled as continuous variables in $\mathbb{R}^{n_i}$. Due to the necessity of interfacing with such continuous observation and action spaces while maintaining the simplest possible structure for efficient learning without an internal model of the vehicle, the model-free algorithm DDPG, developed by Lillicrap et al. (2015), is employed in this work. It belongs to the class of Actor-Critic Methods: the actor approximates the optimal policy deterministically, meaning it always outputs the best-believed action for any given state. The critic, on the other hand, approximates the value function, which is used to critique the actions made by the actor. The so-obtained tuples of state, action, reward, and next state are stored in an experience replay buffer and sampled randomly in a mini-batch to serve as a new dataset in each update iteration.

3.3 Reinforcement Learning Agent

The core of the vehicle control is represented by a DDPG agent that interacts with the environment, as shown in Fig. 2, receiving Observations and computing Actions. To implement LK and ACC functions, the action signal at the generic time instant k is as (1):

$$a(k) = \begin{bmatrix} \dot{v}_r(k) \\ s_{spd}(k) \end{bmatrix}, \tag{1}$$

where $\dot{v}_r(k) \in [-3, 3]$ m/s^2 and $s_{spd}(k) \in [-8, 8]$ rad/s denotes the requested vehicle acceleration and steering speed, respectively.

The Observation set is made up of 35 features mainly describing the vehicle state, the virtual lane in the vehicle reference system (Fig. 3) elaborated by the vehicle logic, and the reference signals received by the infrastructure, as described in Table 1.

Since DDPG is a deterministic agent, during learning, a noise structure $\mathcal{N}$ is superimposed to the actions taken to allow for exploration. It consists of a bi-dimensional Uhlenbeck and Ornstein (1930) process with initial variance $\sigma_0 = [0.3; 4]$, mean $\mu = [0; 0]$, variance decay rate $d_v = 5 \times 10^{-7}$ and minimum variance of $\sigma_{min} = [0.02; 0.3]$.

The Actor and Critic are implemented through two Deep Neural Networks (DNNs). The actor network consists of 4 layer of Fully Connected (FC) layers $[300 \times 600 \times 600 \times 2]$ with activation functions [*ReLu, Lin, ReLu, Tanh*]. The critic network presents two input layers for both Observations and Actions, both followed by an FC *ReLu* activated layer of 600 nodes, that are concatenated and given as input in a following FC *ReLu* layer of equal size. The Q-value is than returned by the output layer as a linear combination of the previous one.

3.4 Virtual Lane Logic

As shown in Fig. 2 the DDPG agent computes the control action exploiting information related to the VL and described by the tuple (θ_{vl}, lo_{vl}, c_{vl}, and p_{vl}), whose definition is provided in Table 1.

The VL data are generated through a set of algorithms that elaborate the following information:

- $p_{lb} = [x_{lb}; y_{lb}]$ coordinates of the points in the vehicle reference system (Fig. 3) at the described distances of each lane boundary of the road section;

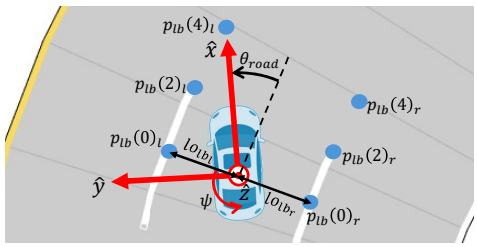

Fig. 3. Vehicle reference system with VL observed data. In brackets, the distance along the road axis, in meters.

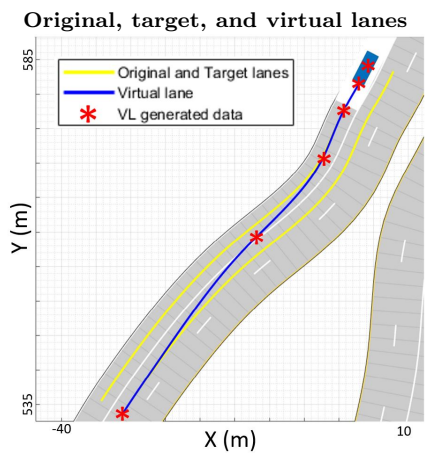

Fig. 4. VL data generated from *lc* curves of l_o and l_t.

Table 1. Observation space features.

Symbol	Domain	Unit	Description
θ_{vl}	$[-180, 180]$	deg	Heading angle error of the vehicle with the virtual lane direction, along the $\hat{z}$ axis of the vehicle reference system.
lo_{vl}	$[-10, 10]$	m	Distance between the center of mass of the vehicle and the center of the virtual lane.
c_{vl}	$\mathbb{R}^6$	deg/m	Curvature of the virtual lane in 6 points at fixed distances from the vehicle calculated along the curvilinear reference frame parallel to the virtual lane center curve.
p_{vl}	$\mathbb{R}^{7\times 2}$	m	x_{vl} and y_{vl} coordinates in the vehicle reference frame of 7 virtual lane center points.
v_{ego}	$\mathbb{R}^2$	m/s	Longitudinal and lateral speed of the center of mass of the vehicle, in its own reference system.
v_{tar}	$\mathbb{R}$	m/s	Cruise speed requested to the vehicle from the infrastructure system.
e_{pos}	$\mathbb{R}^2$	m	Position of the target point to be reached in the vehicle reference frame.
ψ	$[-180, 180]$	deg	Angle of the vehicle around its $\hat{z}$ axis with the fixed north direction.
$\dot{\psi}$	$\mathbb{R}$	deg/s	Angular velocity of the vehicle around the $\hat{z}$ axis.
s_{spd}	$[-8, 8]$	deg/s	Steering speed value of the vehicle.
s	$[-40, 40]$	deg	Steering angle of the vehicle around its $\hat{z}$ axis between the frontal steering wheels and the $\hat{x}$ axis.
$\dot{v}_{ego}$	$\mathbb{R}^2$	m/s^2	Longitudinal and lateral acceleration of the center of mass of the vehicle, in its reference system.
$\ddot{v}_{ego}$	$\mathbb{R}^2$	m/s^3	Longitudinal and lateral jerk of the center of mass of the vehicle, in its reference system.

- c_{lb} curvatures of the lane boundaries in the considered points;
- θ_{lb} heading angle of each lane boundary with respect to the vehicle direction;
- lo_{lb} distance of each lane boundary from the center of mass of the vehicle;

directly extracted from the scenario data, and a set of information provided by the infrastructure defining:

- *original* and *target lanes*;
- λ_{ch} length of the lane change maneuver;
- *offset* initial distance from the starting point of the maneuver.

The *VL data generator* subsystem visible in Fig. 2 extracts from the lane boundaries data the information describing the center curves of the needed lanes (the current in case of LK and both original and target in case of LC), exploiting an algorithm based on averaging right and left boundaries data couples for each ith needed point of the road, following (2), (3), (4), and (5):

$$p_{lc}(i) : \begin{cases} x_{lc}(i) = (x_{lb}(i)_l + x_{lb}(i)_r)/2 \\ y_{lc}(i) = (y_{lb}(i)_l + y_{lb}(i)_r)/2 \end{cases} \tag{2}$$

$$c_{lc}(i) = \frac{2(c_{lb}(i)_l \cdot c_{lb}(i)_r)}{c_{lb}(i)_l + c_{lb}(i)_r} \tag{3}$$

$$\theta_{lc} = \theta_{lb_l} = \theta_{lb_r} = \theta_{road} \tag{4}$$

$$lo_{lc} = (lo_{lb_l} + lo_{lb_r})/2 \tag{5}$$

The extracted lane center (lc) data are directly used as VL data when the vehicle is required to keep its current lane. If a lane change is needed, the VLL exploits a weighted average based strategy to define the LC path. The used weights q^o and q^t of original and target lane points are defined in (6) and (7), where $d \in [0, \lambda_{ch}]$ represent the distance between the point from the start of the maneuver.

$$q^o(d) = \frac{10d^3}{\lambda_{ch}^3} - \frac{15d^4}{\lambda_{ch}^4} + \frac{6d^5}{\lambda_{ch}^5}, \tag{6}$$

$$q^t(d) = 1 - q^o(d). \tag{7}$$

To ensure the regularity of the proprieties of continuous curvature and the derivability of the final VL curve, the LC path has been designed with initial and final angles with the original and target lanes equal to zero. For this reason, the polynomial function in (6) has been designed by imposing a null derivative in the initial and final points of the interval, a null second derivative in the middle point, an initial value equal to zero, and a final one equal to one. The *LC data manager* also exploits the information about the distance w between the two lanes center curves to compute the angle and the curvature of the straightened lane change path that will be later moduled over the shape of the curved road through the weighted average method. The angle $\beta_{ch}(d)$ of the LC path heading on the straightened road, and the curvature $c_{ch}(d)$ computed as:

$$\beta_{ch}(d) = \arctan\left(w\left(\frac{60\,d}{{\lambda_{ch}}^3} - \frac{180\,d^2}{{\lambda_{ch}}^4} + \frac{120\,d^3}{{\lambda_{ch}}^5}\right)\right), \tag{8}$$

$$c_{ch}(d) = \frac{60\,d\,w\left(2\,d^2 - 3\,d\,\lambda_{ch} + {\lambda_{ch}}^2\right)}{{\lambda_{ch}}^5\left(900\,d^4\,w^2\,(d-\lambda_{ch})^4/{\lambda_{ch}}^{10} + 1\right)^{3/2}}. \tag{9}$$

Finally, the *VL data generator*, given the previously computed center curves data and the outputs of the *LC data manager*, can reconstruct the needed LC virtual lane with Algorithm 1. Fig. 4 shows an example of the VL generation during LC, where the shuttle, leaving from the starting station, must merge into the central driving lane on a curved section of the road.

Algorithm 1 Virtual lane data computation

Input data: q^o, q^t, p_o, c_o, lo_o, p_t, c_t, lo_t, θ_{road}, β_{ch}, c_{ch}
Function: Compute the coordinates of virtual lane change for each ith point of distances list $[d_{haed}]$:
for i in $[d_{haed}]$ **do**

$$p_{vl}(i) = q^o(i)p_o(i) + q^t(i)p_t(i) \tag{10}$$

end for
Compute the lateral offset, making use of the first coefficient q_1^t of the vector:

$$lo_{vl} = (1 - q^t(0))lo_o + q^t(0)lo_t\,. \tag{11}$$

Compute the heading angle of the virtual lane ($\langle sch \rangle$ indicates the direction of the lane change):

$$\theta_{vl} = \theta_{road} + \langle sch \rangle \beta_{ch}. \tag{12}$$

Compute the curvature of the VL:
for i in $[d_{haed}]$ **do**

$$c_{vl}(i) = \left(\frac{1 - q^t(i)}{c_o(i)} + \frac{q^t(i)}{c_t(i)}\right)^{-1} + \langle sch \rangle c_{ch}(i)\,. \tag{13}$$

end for
Return: p_{vl}, c_{vl}, θ_{vl}, lo_{vl}

Table 2. Termination conditions (ε_t)

Name	Condition
Crash	$\lvert lo_{vl}\rvert > 1$m
Discomfort	$\lvert \dot{v}_{lat}\rvert > 1.3$ m/s^2
Too slow	$v_{ego} < v_{min}$
Failure	Missed target stop station

3.5 Agent training

The agent has been trained through a curriculum learning methodology. The first phase (on the randomly initialized agent networks) aims at learning a general policy from base-level driving capability. To this aim, it employed the standard Curved Road scenario from the MATLAB Automated Driving Toolbox library, characterized by various curvature radius and directions that reduce the learning bias. Initial and target positions, as well as target and limit speed, have been randomly defined to avoid unwanted correlations between data. The second phase specialized the agent directly on the target scenario (Fig. 1), following 12 fixed routes and speed limits between the shuttle stations. Such a scenario is characterized by additional non-idealities (such as oscillating high values of road curvature, even in road sections that seem straight), resulting in a more complex geometry of lanes.
The reward function $R(k)$ is given in both phases by

$$R(k) = cp\left(1 - (e_p(k)/6)^2\right) - 0.15lo_{vl}^2(k) - 0.01e_s^2(k) + \\ -0.5s_{spd}^2(k-1) - 0.1\dot{v}_r^2(k-1) - 10\varepsilon_t + 3ct + cs. \quad (14)$$

It uses punishment for errors lo_{vl}, e_s, and e_p referred to lateral offset, speed, and distance from the target position. The quadratic costs s_{spd}^2 and $\dot{v}_r^2$ are referred to action's amplitude. Finally, ε_t, ct, cs, and cp represent sparse punishment (the first) and rewards related to catastrophic termination of the episode of Table 2, correct virtual lane tracking ($|lo_{vl}| < 0.1$m), correct speed ($|e_s| < 1$m/s), and approach to the final position ($e_p < 6$m).
Promoting desired behavior has also been forced by employing episode termination conditions as defined in Table 2. While preventing high values of lateral acceleration to allow comfort and safety, the termination system checks for too high lateral deviations and stops accomplishment while ensuring a minimum speed of the vehicle (growing at 0.3 m/s^2 and limited at $0.4v_{tar}$). Each training session consisted of up to $E = 4000$ episodes of a maximum of $E_\ell = 2000$ steps, with a sampling time of 0.05s.

4. SIMULATION RESULTS

As discussed in the previous section, following the curriculum learning technique, the DDPG agent training is divided into two main parts: the first part is performed in the Curved Road scenario (Generalization training), while the second in the Target scenario (Specialization training). Fig. 5 shows the reward associated with each episode over the two training sessions, with the superimposed orange line illustrating its moving average. In the generalization training, the agent starts to record its first successes after about 1500 episodes and finishes after 3500 episodes. The knowledge acquired by the agent during this phase plays a key role in the subsequent session. In fact, since the first episodes of Specialization training, the agent has recorded some successes that speed up this second learning phase.

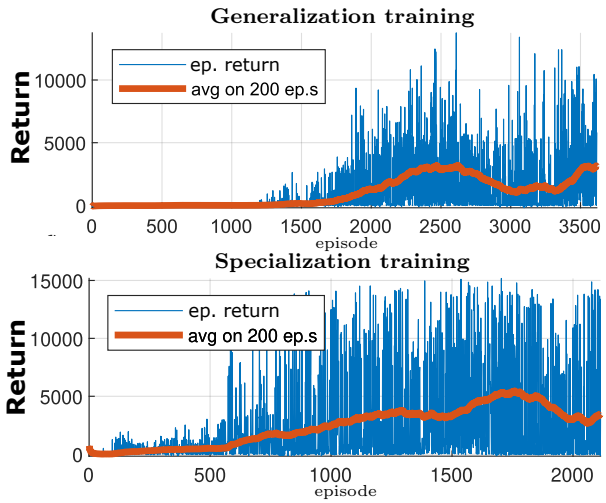

Fig. 5. Return trend in the two training phases.

Before testing the trained agent in an extensive simulation scenario, we assess its capability to overcome potential external disturbances. These disturbances can be caused by unpredictable events such as wind, road irregularities, etc. This evaluation is done by simulating within the Curved Scenario to obtain a more uniform and reproducible test. The error in both heading angle and lateral offset are evaluated here. As shown in Fig. 6, the obtained results remain within an acceptable range even in the presence of uniformly distributed noise that directly affects the control action computed by the DDPG agent.
A comprehensive simulation of the target scenario is carried out to evaluate the effectiveness of the proposed solution. The simulations have been performed in the MATLAB and Simulink environments. The Automated Driving toolbox and the Vehicle Dynamics Blockset have been employed to simulate, respectively, the desired path and the vehicle dynamics. As depicted in Fig. 7, the vehicle begins its journey from the initial station. It then merges onto the main road, travels along the primary curved routes, and finally exits to reach the target destination station. In the graph illustrating the input control action (Fig. 8), it is noticeable that both the vehicle longitudinal acceleration and steering speed are consistently within the design limits ($\dot{v_{ego}} \in [-3; 3]$m/s^2, $s_{spd} \in [-8, 8]$ rad/s) previously established for comfort purpose in Section 2. In addition, the vehicle can accurately track the VL with a maximum error of about 30 cm (Fig. 8 A) while maintaining lateral acceleration (Fig. 8 B) within the design comfort range defined in Table 2 ($|\dot{v_{lat}}| < 1.3$m/s^2). Satisfactory results are also obtained for the vehicle's steering angle (Fig. 8 C) and speed (Fig. 8 D). In particular, the latter shows how the shuttle accelerates smoothly, reaches and maintains the maximum allowed speed of 30 km/h, and finally slows down to stop at the target station.
An animation video of the simulation is available online Fasiello et al. (2024).

5. CONCLUSION

This work presented a new framework for executing the main DDTs required in an AD SAE level 4 shuttle solution. It is based on the exploitation of an RL agent able to perform both ACC and LK, supported by a dedicated VLL that rapidly generates the virtual lane data, enabling the

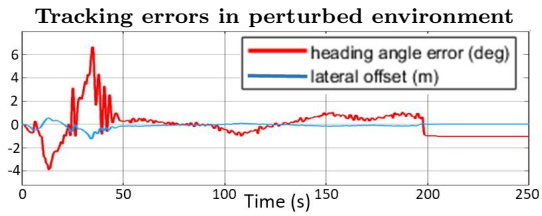

Fig. 6. Heading error and lateral offset of normal disturbances on acceleration and steering speed of variance $\sigma_d = [0.1; 0.2]$.

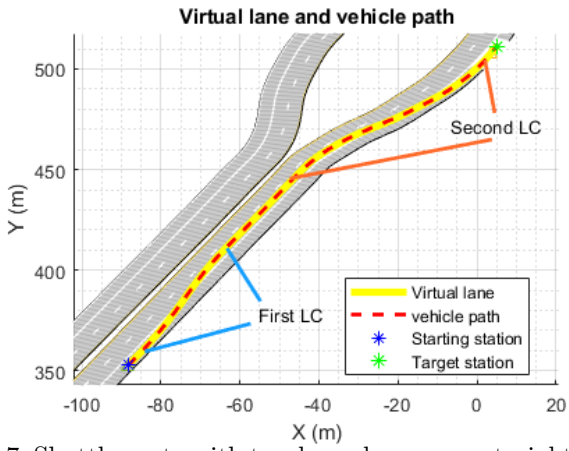

Fig. 7. Shuttle route with two lane changes on straight and curved road sections.

execution of LC maneuvers independently from the road geometry. The approach showed great potential, demonstrating a high ability to control the vehicle, meeting both comfort requirements and vehicle constraints, and overcoming potential disturbances.

The proposed VLL implementation has several advantages over standard RL methods. One of the main is the capability, thanks to VLL, of exploiting less complex network, similar to the ones observed for the racing applications, in the urban scenario, where convolutional and recurrent neural networks are often used. The proposed strategy exploits standard data already available from vehicle onboard safety systems, such as lane boundaries recognition, making it integrable with current technology. This streamlined approach not only makes the training process less complex but also enhances the adaptability of the trained actor, enabling it to handle unforeseen scenarios effectively through a strategic redesign of the VLL. At the same time, compared with traditional control techniques based on online optimization, the proposed solution minimizes the computational effort and delay between data acquisition and control action.

ACKNOWLEDGEMENTS

Computational resources are provided by hpc@polito, which is a project of Academic Computing within the Politecnico di Torino (`http://www.hpc.polito.it`).

REFERENCES

Anzalone, L., Barra, S., and Nappi, M. (2021). Reinforced curriculum learning for autonomous driving in CARLA. In *2021 IEEE International Conference on Image Processing (ICIP)*, 3318–3322. IEEE.

Bengio, Y., Louradour, J., Collobert, R., and Weston, J. (2009). Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, 41–48.

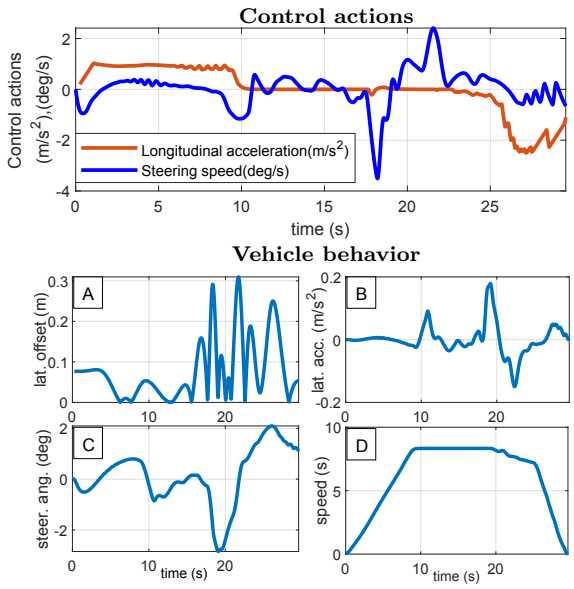

Fig. 8. Action signals and vehicle behavior during the route execution.

Canale, M. and Razza, V. (2024). Automated driving control in highway scenarios through a two-level hierarchical architecture. *IEEE Access*, 12, 86470–86486. doi:10.1109/ACCESS.2024.3416670.

De Winkel, K.N., Irmak, T., Happee, R., and Shyrokau, B. (2023). Standards for passenger comfort in automated vehicles: Acceleration and jerk. *Applied Ergonomics*, 106, 103881.

Fasiello, A., Cerrito, F., Razza, V., and Canale, M. (2024). Enhancing reinforcement learning for automated driving through virtual lane logic. Aviable at `https://youtu.be/rzi4Ckx6j54`.

Haklay, M. and Weber, P. (2008). Openstreetmap: User-generated street maps. *IEEE Pervasive computing*, 7(4), 12–18.

Hua, G., Huang, Z., Wang, J., Xie, J., and Shen, G. (2022). Exploration strategy improved ddpg for lane keeping tasks in autonomous driving. In *Journal of Physics: Conference Series*, volume 2347, 012020. IOP Publishing.

Inc., T.M. (2020). Matlab version: 9.0.0 (R2020b). URL `https://www.mathworks.com`.

Kaelbling, L.P., Littman, M.L., and Moore, A.W. (1996). Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4, 237–285.

Kiran, B.R., Sobh, I., Talpaert, V., Mannion, P., Al Sallab, A.A., Yogamani, S., and Pérez, P. (2021). Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 23(6), 4909–4926.

Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. (2015). Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*.

SAE International (2021). Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles (SAE J3016). URL `https://www.sae.org/standards/content/j3016_202104/`.

Uhlenbeck, G.E. and Ornstein, L.S. (1930). On the theory of the brownian motion. *Physical review*, 36(5), 823.

Wang, Q., Zhuang, W., Wang, L., and Ju, F. (2020). Lane keeping assist for an autonomous vehicle based on deep reinforcement learning. Technical report, SAE Technical Paper.

Wang, S., Jia, D., and Weng, X. (2018). Deep reinforcement learning for autonomous driving. *arXiv preprint arXiv:1811.11329*.

Watzenig, D. and Horn, M. (2016). *Automated driving: safer and more efficient future driving*. Springer.

Zhu, Z., Gupta, S., Gupta, A., and Canova, M. (2024). A deep reinforcement learning framework for eco-driving in connected and automated hybrid electric vehicles. *IEEE Transactions on Vehicular Technology*, 73(2), 1713–1725. doi:10.1109/TVT.2023.3318552.