Optimal Motion Design for Autonomous Vehicles With Learning Aided Robust Control

Attila Lelkó and Balázs Németh, *Member, IEEE*

***Abstract*—This paper presents a control design framework for the integration of robust controller and reinforcement learning-based (RL) control agent. The proposed integration method is applied to motion control of autonomous road vehicles, providing safe motion. The RL-based control agent is used to determine the steering angle and reference velocity of the vehicle to achieve high-performance motion. The chosen reward function is used to achieve different driving behaviors, e.g. high-velocity motion with minimal lap time, path following, or the limitation of control energy. The RL-based control through Proximal Policy Optimization method during episodes is performed. Safe motion is achieved by using a supervisory control framework which is based on the robust $\mathcal{H}_\infty$ control method, and able to keep limits on lateral path tracking error. The effectiveness of the proposed control through simulation examples with comparisons to predictive control methods is illustrated. Moreover, the applicability of the method through a real-life test scenario on a small-scaled test vehicle is demonstrated.**

***Index Terms*—Robust $\mathcal{H}_\infty$ control with learning, autonomous vehicles, motion optimization.**

I. Introduction and Motivation

VARIOUS performance specifications are posed against autonomous vehicles, which must be guaranteed by the control systems. Usually, there are primary performance specifications, which must be kept due to safety reasons, such as guaranteeing vehicle motion stability, reliability, or keeping different traffic regulations. Moreover, several further non-safety performance specifications can be defined, which have lower priority, e.g., providing passenger comfort, achieving economic motion, minimization of traveling times, etc.

In recent years various solutions to guarantee performance requirements of autonomous vehicles with control design have been proposed. One group of the solutions is to enhance the achieved performance level using the data-driven extension of the classical control tools, e.g., Model Predictive Control (MPC, see [1], [2], [3]), model-free control (MFC, see [4]), robust and Linear-Parameter Varying (LPV, see [5], [6], [7]) methods. Using these methods enhanced performance levels on comfort and energy consumption can be achieved, and similarly, the safety performance level of the autonomous vehicles might also be handled [8]. Robust control tools have high relevance in the context of autonomous vehicles, as various noises, disturbances, and unmodeled dynamics may arise during vehicle motion. The impacts of these unwanted effects can be handled using robust design methods, e.g., in [9] a robust controller has been designed in a Driver-in-the-Loop scenario. In paper [10] robust driver assistance system for handover scenarios has been demonstrated, or [11] has proposed an MPC solution for achieving rollover prevention. The classical vehicle control solutions may have the disadvantage that generally they are based on simplified control-oriented models that result from dynamic relationships. Although robust vehicle control synthesis techniques can handle some types of uncertainties they might fall behind the data-driven methods in case of high-level performance requirements.

Using a large number of measurement data on the system and applying data-based adaptive methods leads to another group of solutions. Addressing performance problems involves the implementation of learning-based techniques, particularly those utilizing neural networks and deep learning, within a control framework. The advantage of these methods is that the neural network can be trained with a large amount of data obtained from the actual operation of the vehicle, thus achieving the optimal solution of the vehicle control problem [12]. The effectiveness of neural networks in control applications has been investigated in different studies, e.g., [13], [14]. A survey on deep learning applications in vehicle control has been presented by [15]. Nevertheless, a disadvantage of using neural-network-based methods in safety critic applications is the lack of theoretical guarantees on safety performances. Neural networks are usually highly complex systems designed by a numerical optimization using a finite set of data of the problem, and the result of the training is rated statistically. All possible scenarios cannot be included in the training set and the generalization of the network is hard to validate. This challenge motivated research to provide analysis techniques on the performance evaluation of neural-network-based control systems, e.g., papers [16], [17], [18] focus on the verification of the designed networks. Despite the achieved results, the developed solutions are valid for special cases concerning the control problem or the structure of the neural network, i.e., limiting their applications in general.

Manuscript received 9 October 2023; revised 10 February 2024 and 2 April 2024; accepted 10 April 2024. Date of publication 1 May 2024; date of current version 19 September 2024. This work was supported in part by the European Union within the framework of the National Laboratory for Autonomous Systems under Grant RRF-2.3.1-21-2022-00002, and in part by the National Research, Development and Innovation Office (NKFIH) through OTKA under Grant K 135512. The review of this article was coordinated by Dr. Zhihong Yao. *(Corresponding author: Attila Lelkó.)*

The authors are with the Institute for Computer Science and Control, Hungarian Research Network, H-1111 Budapest, Hungary, and also with the Department of Control for Transportation and Vehicle Systems, Faculty of Transportation Engineering and Vehicle Engineering, Budapest University of Technology and Economics, Műegyetem H-1111 Budapest, Hungary (e-mail: attila.lelko@sztaki.hu; balazs.nemeth@sztaki.hu).

Digital Object Identifier 10.1109/TVT.2024.3394699

The disadvantages of the previous two groups of solutions have led to the development of integrated methods, in which classical model-based control techniques and learning-based approaches are incorporated simultaneously. The integration aims to combine these two solutions to achieve the high performance of learning-based methods, and also the robustness and reliability of classical techniques [5]. The integration on various levels of autonomous vehicle control can be achieved. Advanced vehicle modeling frameworks have been developed, which involve data processing on the step of model formulation, e.g., through closed-loop matching [19]. Focusing on the step of control design, it has been provided design frameworks, in which classical and learning-based control solutions jointly have been involved, e.g., robust [20] and LPV control with neural networks [21], or [22] has proposed a safe model-based reinforcement learning method to achieve control for LPV systems. Furthermore, data can be incorporated in the control design for coordinating multiple unmanned vehicles [23]. The benefits of data-aided control on this level are reduced energy consumption or transportation network load [24], [25].

The brief literature overview shows that lots of efficient results in the integration of classical and learning-based methods have been achieved. Nevertheless, a limitation of the results is that most of them have fixed structures on the control design or the learning process. Since modularity is a crucial aspect of control design for autonomous vehicles (see e.g., [26], [27]), the control design method is proposed to employ independent design of both learning-based and robust control. This concept achieves the integration of the two controllers through a supervisor [5]. This paper aims to present an integrated vehicle control strategy based on the previous concept, with which longitudinal and lateral controls are designed. For achieving high-performance motion of the vehicle, reinforcement learning (RL) is used. For guaranteeing safe motion, robust control based on the $\mathcal{H}_\infty$ method and a supervisor based on quadratic programming are used. The result of the design process is a control system, which calculates both longitudinal and lateral control inputs of the vehicle.

The control design process is presented to optimize the motion of the autonomous vehicle, focusing on minimizing lap time. This problem has been chosen because there are several methods available for its solution, which provide a basis for comparison. For example, [28] investigates the effect of all-wheel drive on achievable lap time via convex optimization. Different optimal control methods for achieving minimum lap time are compared in [29], and a fast Bayesian optimization is shown in [30]. These methods are for offline calculation on a given track to be used for path tracking. Learning Model Predictive Control (LMPC) solution for small-scaled test vehicles is proposed in [3]. The aim of LMPC is to learn the terminal cost and terminal set during the motion of the vehicle for achieving enhanced motion capabilities. Paper [31] proposes a Model Predictive Contouring Controller (MPCC) solution with Gaussian Process to control miniature race cars and achieve significant lap time reduction compared to a baseline controller. In this solution, the lap time minimization is built into the MPCC optimization problem, and computes the optimal interventions online, and it is able to adapt to different racetracks. Although MPC-based solutions can improve lap time efficiency, their disadvantage is the increased computation time in the cases of nonlinear vehicle models and large prediction horizons. Consequently, in this paper a novel solution to the problem of lap time minimization is provided. The contributions of the paper to the state-of-the-art solutions and the existing own preliminary solution [32] are twofold. First, the training of learning-based and the design of model-based controllers are independent processes, but their results are built into the control structure simultaneously. Second, the effectiveness of the integrated control through comparative simulations and demonstrations on a small-scaled indoor test vehicle is presented.

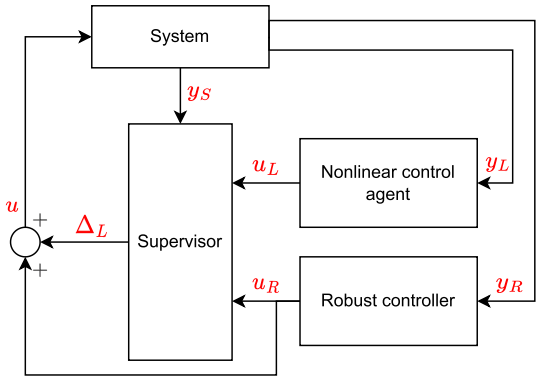


Fig. 1. Scheme of the control architecture.

The paper is organized as follows. The concept of the learning-aided robust control is introduced in Section II. The design process of the RL-based control design, together with the formulation of the applied vehicle model is found in Section III. The design of the robust $\mathcal{H}_\infty$ control and the computation of the safe velocity profile are presented in Section IV. The design of the supervisor for connecting RL-based control and robust control is proposed in Section V. That section also contains the optimization process of the control loop for achieving enhanced performance level. In Section VI the proposed control method is evaluated, i.e., the enhancing process of the control loop and comparisons for existing MPC-based solutions are found. Section VII details the process of control implementation on a small-scaled test vehicle, together with the demonstration of a test scenario. Finally, the paper is summarized and the future challenges are proposed in Section VIII. Moreover, at the end of the paper, Table III contains the list of notations.

II. Overview on the Concept of Learning-Aided Robust Control Design

The goal of this overview is to briefly introduce the concept of our proposed robust control design method. The scheme of the control architecture can be found in Fig. 1. The architecture contains three main elements, such as the robust control, the supervisor, and the reinforcement learning (RL) based agent. The role of the robust control and the supervisor is to guarantee safety performance requirements, i.e., these two elements together guarantee a functionality, which is similar to the safety

filter. The role of the RL-based agent is to improve the safety and non-safety performance levels of the control system, which improvement is achieved through a training process. During the operation of the system, all of these elements operate simultaneously, and the control input (u) is computed through the output of the supervisor (Δ_L) and the output of the robust control (u_R), such as

$$u = u_R + \Delta_L, \tag{1}$$

where $\Delta_L \in [\Delta_{L,\min}; \Delta_{L,\max}]$ is a peak-bounded addition to u_R.

The idea behind the concept is as follows. The output of the RL-based controller (u_L) is considered to be a candidate control input of the system. A dynamical representation of the system in the design of the robust control is incorporated, which results in the reference control input u_R. The reason of robustness is that u is not necessarily equal to u_R, and thus, $[\Delta_{L,\min}; \Delta_{L,\max}]$ domain of Δ_L addition in the design of the robust control is considered as an uncertainty. Using u_L, u_R and measurements on the system (y_S), the supervisor has to select Δ_L. This selection leads to a constrained optimization task. The objective of the optimization is to minimize $\|u(\Delta_L) - u_L\|_2^2$, i.e., u must be as close as possible to u_L for achieving high performance level in the operation. Nevertheless, this selection is constrained by the bounds of Δ_L, which means that u is in the predefined neighborhood of u_R. Moreover, the selection can also be constrained by further criteria, e.g., the predicted tracking error, resulting by u.

The design of the proposed control architecture has the following steps. First, the robust control with predefined bounds on Δ_L is designed. Second, the optimization process in the supervisor is formulated. The optimization during the operation of the control system is solved in each step. Third, the environment for the RL process is constructed, which environment contains the designed robust control, the supervisor, and the system itself. Fourth, the RL process using the constructed environment is performed. The resulting neural network after the learning process can be used, and thus, the requested computation time is reduced during the operation of the control system.

The proposed control concept has some connections to the MPC-based safe learning approaches, which connections provide motivation for comparison to different existing solutions. The most important connection is the formulated constrained optimization task in the supervisor. Despite the MPC method, this optimization does not contain terminal set and terminal cost terms. Their roles are built in the robust control, which provides the reference control input. Thus, the safe set of states is determined by the controllability set of the robust control, which set is further constrained by the supervisor.

From the other side, the learning task is also out of the supervisor, because this task is involved in the RL-based control. Consequently, it may require reduced real-time computation effort compared to methods relying mostly on MPC with complex nonlinear models, which is supported by the performed tests on the control framework. Moreover, the separation of different tasks into different elements, such as learning and guaranteeing safety, can provide a modular architecture. It may provide the possibility of replacing the RL-based controller or the robust controller (e.g., through an updating process) without a comprehensive redesign process on the entire control loop. Modularity also provides the possibility of separating physically each element, e.g., the RL-based agent can be implemented on the cloud, while the supervisor and the robust control on the physical device can be found [33]. Another benefit of the method is that the design of the robust control and the formulation of the supervisor are independent of the internal structure of the RL-based agent. Consequently, other types of agents can also be used, e.g., supervised learning [21]. Therefore, the proposed control structure can be compatible with the application of machine-learning-based techniques, which compatibility may increase the number of application areas of the proposed method, e.g., y_L can contain video frames.

III. Design of RL-Based Agent for Autonomous Vehicle Control

In this section, the design method of the learning-based controller is detailed for handling longitudinal and lateral dynamics. First, the vehicle model is formulated for control purposes and second, the training process is presented. Third, the impact of reward function selection on the achievable performance level of the learning-based control is analyzed.

A. Formulating Vehicle Model for Integrated Control Purposes

The design of motion control for autonomous vehicles requests the formulation of their dynamic models. The model has three purposes in the integration, and thus, it depicts the validity of the control operation, i.e., inside of the validity of the selected vehicle model. First, the motion model is used for the training of RL-based agents, i.e., it serves as a part of the learning environment. Second, the motion model can have a crucial role in the design of the robust control, due to its model-based property. Third, it can be built in the supervisor to form vehicle-safety-oriented conditions in the constrained optimization problem. All of these tasks require a limited complexity vehicle model, i.e., to avoid insufficiently long training process, numerical difficulties in the robust control design, or slow real-time running performance in the solution of the supervisory optimization process. Therefore, in this paper, a simplified dynamical two-wheel dynamical motion model is formed [34]. In the cases of robust control and supervisor design, the formulated vehicle model contains linear tire-force characteristics, such as:

$$m\dot{v}_x = F_{drive} - bv_x + mv_y\dot{\psi}, \tag{2a}$$

$$m\dot{v}_y = -C(\alpha_F + \alpha_R) - mv_x\dot{\psi}, \tag{2b}$$

$$\Theta_z\ddot{\psi} = -C(\alpha_F + \alpha_R)\frac{l}{2}, \tag{2c}$$

where F_{drive} is the driving force, b is a coefficient of friction in the longitudinal velocity, C is the cornering stiffness, α_F and α_R are tire side-slips at the front and rear tires respectively, and L is the length of the wheelbase. Tire side-slip angles and v_y lateral velocity can be expressed as the functions of yaw rate $\dot{\psi}$, steering angle δ, and v_x. Relations in (2) can be used in the

robust control design and within the supervisory optimization, resulting in local velocities in the frame of vehicle reference and the yaw angle ψ, which can be integrated to estimate the path tracking error. The longitudinal and lateral velocity components of the vehicle in a global coordinate system are calculated as

$$V_x = v_x \cos\psi - v_y \sin\psi, \tag{3a}$$

$$V_y = v_x \sin\psi + v_y \cos\psi, \tag{3b}$$

which can be used for computing the position of the vehicle through the integration of V_x, V_y.

For learning purposes, it is possible to slightly improve the complexity of the model, i.e., steering dynamics and nonlinear tire-force characteristics are formulated. Its reason is that the training process is carried out offline, and thus, the nonlinear model is not required to run during the control process. The dynamics of the steering mechanism a simple low-pass filter is introduced on the steering angle:

$$\dot{\delta} = \frac{1}{T_\delta}(\delta_{ref} - \delta), \tag{4}$$

where δ_{ref} is the reference steering angle signal, and T_δ is the steering model parameter for scaling its time response characteristics. Moreover, the model for the learning process is improved with the Pacejka tire-force model [35]. The tire force is calculated as

$$F_{lat} = F_z \cdot D_t \sin(C_t \text{atan}(B_t\alpha_i - E_t(B_t\alpha_i - \text{atan}(B_t\alpha_i)))), \tag{5}$$

where F_z is the vertical force acting on a specific tire, B_t, C_t, D_t and E_t are constant tire model parameters and α_i is the tire lateral slip angle on the front or rear axle. Lateral tire force formulation (5) is built in the simulation environment that is used for training and control design purposes.

B. Training Process of the RL-Based Agent

The goal of the training process is to design a neural network agent for control purposes, with two outputs corresponding to the reference velocity and the steering angle of the vehicle. The training process is performed in an RL-based framework using the Proximal Policy Optimization (PPO) algorithm.

The target of the training process is to maximize a predefined reward function, during the interaction of the agent with its dynamic environment through actions. The action of the agent is determined by sampling from a probability distribution, i.e., policy $\pi_\theta(\cdot|s)$. The policy is represented by an actor neural network with parameter vector θ and the observed state of the environment s. Every time step t the reward is computed as a function of the environment state and of the action, chosen by the agent $R(s_t, a_t)$, where s_t is the environment state and a_t is the chosen action at time step t. Given a state-action trajectory $\tau_t = (s_t, a_t, s_{t+1}, a_{t+1} \dots)$, the finite-horizon discounted return is the discounted sum of the rewards, collected by the agent starting from state s_t at time step t and following the policy:

$$\mathcal{R}(\tau_t) = \sum_{k=0}^{M-1} \gamma_{\text{d}}^k R(s_{t+k}, a_{t+k}), \tag{6}$$

where γ_{d} is a discount factor. It is used to exponentially decrease the importance of future rewards compared to the present reward along a horizon with M steps.

Another term in the training is the value of a state, which is the expected return starting from state s_t and acting according to the policy:

$$V(s_t) = \mathbb{E}\left[\mathcal{R}(\tau_t)\right], \tag{7}$$

where the expected value is required, if the environment or the policy contains stochastic elements. In the PPO algorithm, the value function is represented by an independent critic neural network, which is trained separately using past experience on the achieved returns. The expected value of the states is calculated based on the information gathered in previous episodes of the training. After a predetermined number of episodes or time steps, the achieved τ_t trajectories are collected. Using this information the return values are calculated by discounting the rewards corresponding to each trajectory, see (6). If different values are achieved starting from the same state (e.g., due to the stochastic nature of the environment or the agent), the mean of these values is calculated.

The agent learns through interactions with the environment. During these interactions different actions are taken, different environment states are observed and rewards are collected. The goodness of an action is determined relative to past experiences using the advantage of that action. The advantage of action a_t describes how much better is to take that specific action in state s_t compared to acting corresponding to the policy. The advantage function $A(s, a)$ determines the advantage of action a in the state s.

In practice, it can be difficult to determine the exact value of the advantage function, but different methods are available for estimation purposes, e.g., Generalized Advantage Estimation in [36]. After simulating the environment for T time steps, the estimation is as follows:

$$\hat{A}(s_t, a_t) = \sum_{k=0}^{T-t-1} \left(\gamma_{\text{d}}^k R(s_{t+k}, a_{t+k})\right) + \gamma_{\text{d}}^{T-t} V(s_T) - V(s_t). \tag{8}$$

The first two terms of this expression are the discounted return starting from state s_t based on the collected experiences, and the third term is the estimated value of state s_t. In this way, the advantage is estimated by the realized and the expected return of state s_t. The value of $V(s_t)$ is computed by the critic network, which is trained using past experiences. If during the simulation a state-action sequence is found where $\hat{A}(s_t, a_t) > 0$, then this sequence is considered to be better expected. Consequently, the probability of such sequences appearing during further interactions should be increased, because they lead to larger rewards.

During the training iterations, the parameter vector of the actor network is constantly changing as a result of the parameter updates. This leads to different policy functions for every update. The ratio of the probability of action a_t using the actual and the old policy is denoted by:

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}. \tag{9}$$

The goal is to increase the probability of actions with positive advantage and to decrease the probability of actions with negative advantage. This goal is expressed using the following surrogate objective function:

$$L(\theta) = \hat{\mathbb{E}}_t \left[r_t(\theta) \hat{A}(s_t, a_t) \right], \tag{10}$$

where the expected value is related to the different state-action trajectories starting from state s_t. The problem with the surrogate objective function is that without constraints it would lead to excessively large policy updates. This is addressed by Trust Region Policy Optimization (TRPO) by limiting Kullback-Leibler divergence [37] or the PPO method by using a clipped surrogate objective function [38]. The algorithm described in this paper uses the PPO method in the optimization. The reason behind this selection is its fast training capability compared to TRPO [38]. The clipped surrogate objective function in the case of the PPO algorithm is formed as:

$$L^{PPO}(\theta) = \hat{\mathbb{E}}_t \left[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t) \right], \tag{11}$$

where ϵ is a hyperparameter of the algorithm. The above objective uses the simplified clip() function to avoid large policy updates, this way it is significantly faster to compute a policy update.

Improvement of the performance level by the RL-based agents can be achieved through reward functions. At every step of the environment, the reward is calculated, based on the reward function of the agent. The simple parametric reward function in the vehicle control task is formed as

$$R(s, a) = -A x_{\text{Lat,err}}^2 - B \delta_{ref}^2 + \Delta p, \tag{12}$$

where $x_{\text{Lat,err}}$ is the lateral path tracking error, Δp is the progress along the centerline since the last time step using an appropriate metric. During training, if the vehicle leaves the track a large punishment through reward functions is applied, but the current episode is not terminated. Thus, the agent can experience situations, which are considered potentially dangerous in a real-life scenario and may learn to solve these situations by navigating back to the track as fast as possible.

The training process is performed through simulation episodes, in which the vehicle must move on given tracks. During the training process, the motion of the vehicle model along various tracks is simulated, together with the consideration of the actual learning-based agent. The closed tracks are generated from track primitive elements, such as straight sections and bends with different curvatures. In every training scenario, Gaussian noise is added to the agent observations to address the uncertainty of the real-life positioning system. This noise is scaled with the distance from the vehicle, resulting in larger uncertainties of track points farther ahead. The expectation against the training process is that the long-term reward is increased.

The inputs of the networks consist of measurements on the track in the neighborhood of the vehicle, which is the position of N number of equidistant points of the track centerline ahead. This results in an input vector of

$$x_{NN} = [x_1 \quad y_1 \quad x_2 \quad y_2 \quad \dots \quad x_K \quad y_K]^{\text{T}}. \tag{13}$$

where the coordinates are in the local coordinate system of the vehicle, x denotes the local longitudinal direction (forward), y is the local lateral direction (left), and the indices correspond to the order of the points. Although the selection of K for a long horizon can result in highly efficient control intervention due to the lots of information on the track, it can overcomplicate the neural network and consequently, training time is significantly increased, and extracting high definition information on the track on long distances can be challenging in real-life applications. Therefore, K is recommended to be selected depending on the bend curvatures on the track, i.e., the requested minimum look-ahead distance, which determines the actual selection of δ and v_x, considering the measurement method used to estimate the points of the centerline.

In the training environment, several noises and disturbances are included to increase the achieved generality and robustness of the trained agent. Additional action noise is included which is a required part of the training process. The estimation of the centerline has uncertainty in a real application, i.e., it is modeled by a noise added to the neural network input, which is proportional to the distance from the vehicle. Communication and hardware delays are modeled into the environment via a constant time delay of the control input. Additionally, the parameters of the vehicle model change slightly during the training process.

C. The Effect of Parameter Selection in Reward Functions

The driving behavior of the agents is significantly influenced by the weights of the reward function. The most typical example is if one chooses weight A large, then the result will be an agent that follows the center of the track accurately. But, if this weight is small compared to Δp, then faster progress can be more important and the agent tends to cut corners aggressively and try to find the ideal path to reduce lap time. Increasing B weight can help avoid a larger steering angle than necessary, resulting in fewer oscillations, more stable motion, and a decrease in the required control energy.

To demonstrate the effect of reward parameters some training examples have been carried out. Fig. 2(a) shows the convergence of the cumulative reward function in selected training processes of the RL-based lateral control. The illustration shows that maximization of the reward can be reached, independently from the selection of parameter A. The convergence of the reward is achieved at 10^6 time steps, which has the computation time request $24-30$ min in this example.

An example of the effect of reward function parameter A can be seen in Fig. 2(b). The graphs show three agents trained with different reward functions, in every case $B = 0$, only the path tracking capabilities and the faster lap time were considered in the optimization process. The difference between the agents can be seen especially in the corners, where the agents with a smaller A weight tend to cut the corners more aggressively. The maximal lateral errors were 0.75 m ($A = 0.01$), 0.54 m ($A = 10$), and 0.26 m ($A = 30$). The time to complete the lap was 12.5 s ($A = 0.01$), 12.85 s ($A = 10$), and 13.55 s ($A = 30$). These results are in accordance with the selection of the reward functions. Fig. 2(c) shows the reference steering

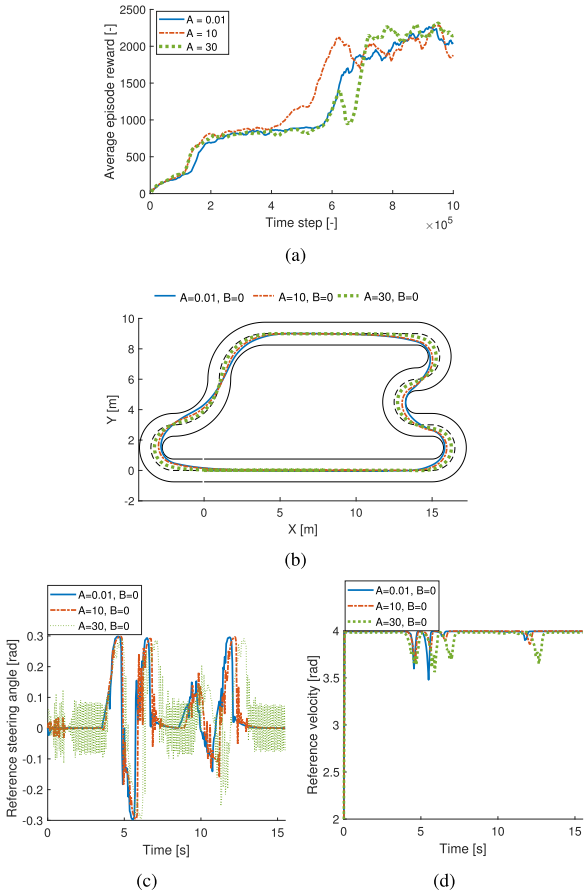


Fig. 2. Convergence of the accumulated reward in selected training processes (a). The effect of A reward function parameter on (b) the resulting trajectory, (c) the reference steering angle, and (d) the reference velocity, in case of $B = 0$.

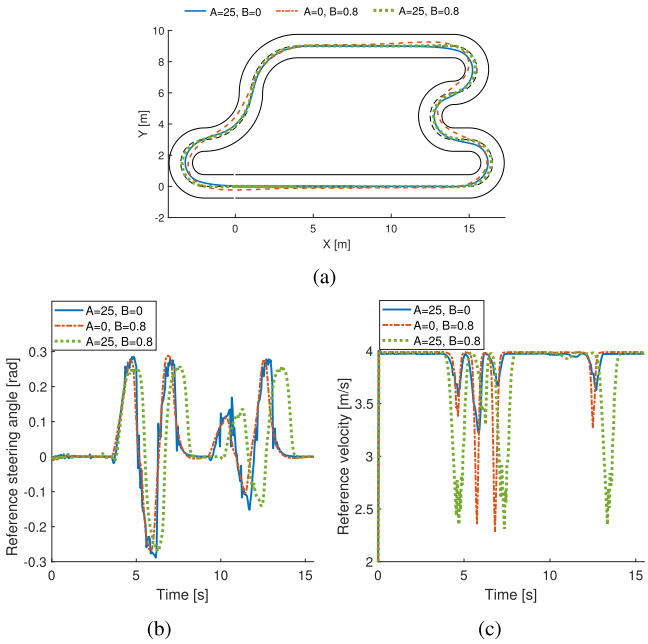


Fig. 3. The effect of B reward function parameter on (a) the resulting trajectory, (b) the reference steering angle, (c) the reference velocity, in case of $A = 0.001$.

angle of the vehicle during the simulation. It can be seen that high-amplitude and high-frequency oscillations are present in the signals, especially in the case when A is high. These are mostly unwanted control interventions that result in oscillatory or even unstable motion in a real-life scenario with very low comfort for possible passengers. Fig. 2(d) indicates that a more conservative reference velocity is required to increase tracking performance.

The unwanted oscillations in the steering control intervention can be effectively eliminated by a related punishment term in the reward function. The effect of such a term is illustrated in Fig. 3. In the examples, the effect of increasing weight B is shown in the case of two different A values. The most significant difference can be seen in Fig. 3(b), where the agents have been trained using the additional punishment term for avoiding sudden changes in the steering control signal. When cornering or path correction is not required, then the steering angle is chosen to be close to zero. In this case, the maximal lateral errors are 0.29 m ($A = 25, B = 0$), 0.66 m ($A = 0, B = 0.8$), and 0.21 m ($A = 25, B = 0.8$), and the lap times are 13.55 s ($A = 25, B = 0$), 13.4 m ($A = 0, B = 0.8$), and 14.6 s ($A = 25, B = 0.8$). Its reason is that with the punishment term B the lap time is relatively less important. Thus, the less aggressive corner-cutting vehicle motion decreases the expected reward of the agent. Larger B values may lead to a slightly decreased reference velocity profile to complete maneuvers with smaller steering angles, see lines blue and green. Nevertheless, without the introduction of the term B, the resulting behavior may not be beneficial in real-life scenarios, because the impact of delays and uncertainties may cause performance degradation.

In Fig. 3 can be seen that by increasing even one of the two punishment terms the lap times start to increase, since progressing along the track will have relatively less effect on the long-term reward of the agent. There is another limiting factor on choosing large weight values because as the punishment terms start to increase the overall reward will decrease and there will be a point (negative long-term reward values) where not moving at all will be more beneficial regarding the cumulative reward, since it results in 0 reward compared to any negative value. It leads to a limit on the minimal lateral error with which path tracking can be performed and a limit on the minimal steering angle values. These training processes may result in unwanted control policies. Consequently, the selection of parameters A and B is a tuning problem in practical applications. During the tuning process, the motion of the real vehicle must be analyzed. Properly chosen A and B values can lead to a reward function that results in a high-performing agent considering path tracking, lap time, and control energy. The presented simulation-based examinations provide references on the influences of terms A and B to the vehicle motion.

IV. Design of the Robust Element of the Autonomous Vehicle Control System

The design of the robust control is based on the method presented in this section. In the control design, it is necessary

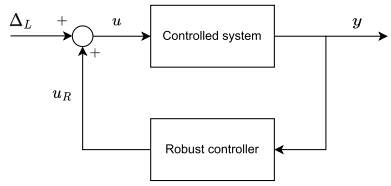


Fig. 4. Schematic view on the loop for robust control design.

to consider that u may differ from u_R, due to the additional value of Δ_L. Therefore, Δ_L can be handled as an additive input disturbance to u_R, and thus, robustness against Δ_L must be guaranteed. The block diagram of the system in Fig. 1 can be restructured to form a simple control loop with additive input disturbance, see Fig. 4. Thus, from the viewpoint of the robust controller, the internal structures of the supervisor and the learning-based control agent are not considered. These are represented by the disturbance term Δ_L.

In the robust control design process, the worst-case scenario is considered, i.e., Δ_L in the robust design process through its bound is involved. In the design is assumed that $L_\Delta \triangleq |\Delta_{L,max}| = |\Delta_{L,min}|$, i.e., the measure of additive input disturbance is symmetric. Since L_Δ has an impact on the robust control design, it influences the values of $\Delta_{L,max}, \Delta_{L,min}$ in the optimization process of the supervisor. Choosing L_Δ to have a large interval allows increased differences in the two control signals, resulting in $\Delta_L = u_L - u_R$ often can be selected. Nevertheless, it results in a more conservative robust controller to provide robust stability even in case of larger disturbances. But, if L_Δ is tight, u is close to u_R, and thus, performance level increase from the RL-based control agent can be lost.

A. Design of Robust Lateral Controller

To design a robust control, several methods are available, with which theoretical guarantees against bounded disturbances can be guaranteed. In the rest of this section a robust $\mathcal{H}_\infty$ control design method on the lateral dynamics, considering Δ_L is proposed. The design is based on the vehicle motion model (2), which can be transformed into the following state-space representation:

$$\dot{x} = \mathcal{A}x + \mathcal{B}_2 u = \mathcal{A}x + \mathcal{B}_2\Delta_L + \mathcal{B}_2 u_K, \tag{14}$$

where $\mathcal{A}, \mathcal{B}_2$ are matrices of the system, $x = [v_y \quad \dot{\psi}]^T$ represents state vector and $u = \delta$ steering angle.

The primary, i.e., the safety performance of the system is to guarantee the limitation of the lateral error of the vehicle from the centerline of the road:

$$z_1 = y_{ref} - y; \qquad |z_1| \to min, \tag{15}$$

where y_{ref} is the reference lateral position for the vehicle. Moreover, the limitation of the steering angle is requested to avoid the unwanted effect of actuator saturation, which leads to further performance:

$$z_2 = u = u_K + \Delta_L; \qquad |z_2| \to min. \tag{16}$$

The performance vector $z_K = [z_1 \quad z_2]^T$ using the state-space (14) can be expressed as

$$z_K = \mathcal{C}_2 x + \mathcal{D}_{22} w + \mathcal{D}_{22} u_K, \tag{17}$$

where $w = [y_{ref} \quad \Delta_L]^T$. Similarly, the formulation of measurement $y_K = y_{ref} - y$ is expressed as

$$y_K = \mathcal{C}_1 x + \mathcal{D}_{11} w + \mathcal{D}_{12} u_K. \tag{18}$$

The control-oriented state-space representation of the system from the dynamics, performances, and measurements on the system is composed, such as

$$\dot{x} = \mathcal{A} + \mathcal{B}_2\Delta_L + \mathcal{B}_2 u_K, \tag{19a}$$

$$y_K = \mathcal{C}_1 x + \mathcal{D}_{11} w + \mathcal{D}_{12} u_K, \tag{19b}$$

$$z_K = \mathcal{C}_2 x + \mathcal{D}_{22} w + \mathcal{D}_{22} u_K. \tag{19c}$$

In the design process of the $\mathcal{H}_\infty$ controller weighting function for scaling disturbances and for finding the balance between different performances must be applied, see [39] for details of selecting weighting functions and the formulation of the $\mathcal{H}_\infty$ design problem. Using the weighting functions and the dynamic controller-observer, represented by $\mathcal{A}_K, \mathcal{B}_K, \mathcal{C}_K, \mathcal{D}_K$ matrices, the closed loop system can be formulated as

$$\dot{x}_{cl} = \mathcal{A}_{cl} x_{cl} + \mathcal{B}_{cl} w, \tag{20a}$$

$$z = \mathcal{C}_{cl1} x + \mathcal{D}_{cl1} w, \tag{20b}$$

where w involves also Δ_L of (19a). The objective of $\mathcal{H}_\infty$ control is to minimize the inf-norm of the transfer function $T_{z_\infty w}$. More precisely, the problem can be stated as follows [40], [41]. The linear matrix inequality (LMI) problem of $\mathcal{H}_\infty$ performance is formulated as the closed-loop RMS gain from w to z_∞ does not exceed $\gamma > 0$ if and only if there exists a symmetric and definite positive matrix X_∞ such that

$$\begin{bmatrix} \mathcal{A}_{cl}X_\infty + X_\infty \mathcal{A}_{cl}^T & X_\infty \mathcal{B}_{cl} & \mathcal{C}_{cl}^T \\ \mathcal{B}_{cl}^T X_\infty & -\gamma I & \mathcal{D}_{cl}^T \\ \mathcal{C}_{cl} & \mathcal{D}_{cl} & -\gamma I \end{bmatrix} < 0. \tag{21}$$

During the robust control design, the value of γ must be optimized, and as a constraint, the formed LMI (21) must be feasible. The result of the optimization is the robust controller-observer, whose robust stability against w can be guaranteed.

B. Computation of Safe Velocity Profile

The computation of the safe velocity profile, i.e., the actual reference velocity is determined by the local curvature of the track. Since the reference path y_{ref} (e.g., centerline) of the track is considered as a known curve $c(s)$ parameterized by the travel distance s, its curvature can be calculated as

$$\kappa(s) = \frac{|\dot{c}(s) \times \ddot{c}(s)|}{|\dot{c}(s)|^3}. \tag{22}$$

The reference velocity is calculated in a way that limits the required lateral acceleration of the car for traction reasons:

$$v_{ref}^2 \kappa \leq a_{y,max}, \tag{23}$$

where $a_{y,max}$ denotes the maximum achievable lateral acceleration, based on the maximal tire force available. Its value can be determined by tire characteristics or estimation, see e.g., [42]. The resulting reference velocity must be chosen as

$$v_{ref} \leq \sqrt{\frac{a_{y,max}}{\kappa}}. \tag{24}$$

In straight sections, the curvature of the track is 0, which results in infinite reference velocity. Thus, v_{ref} must be limited, especially in case of low curvature values, such as $v_{ref} \leq v_{max}$, where v_{max} is the maximum velocity limit on the given road section.

The centerline of the track is considered here as a two-dimensional spline, however for practical reasons in simulations or tests it is represented as a list of equidistant (x, y) coordinates, and the curvature is estimated using numerical differentiation instead.

V. Design of the Supervisor for Guaranteeing Safe Vehicle Motion

The supervisor results in Δ_L signal, which is used for the computation of control input u, see (1). The goal of the supervisor is to achieve a control signal, which results in a high-performance level for the vehicle through the minimization of the following objective:

$$\|u(\Delta_L) - u_L\|_2^2 = \|u_R + \Delta_L - u_L\|_2^2 \rightarrow \min, \tag{25}$$

where $\Delta_L \in [\Delta_{L,min}; \Delta_{L,max}]$ is considered and $\Delta_L = [\Delta_{L,Lat} \quad \Delta_{L,Long}]^T$ contains the additional disturbance value respect to lateral and longitudinal controls.

The control design problem with the vehicle-safety-oriented performance requirement on keeping lateral error under a predefined value is augmented. Thus, in the case of the vehicle path tracking control, the lateral distance from the centerline can be limited using a model-based prediction layer in the supervisor. The trajectory of the vehicle can be predicted on a time horizon using (2)–(3), see [5], which results in the vector of predicted $P_{pred,T}$ vehicle positions. The lateral error of the vehicle can be calculated as

$$e_{lat,T} = dist\left(P_{pred,T}, c(s)\right), \tag{26}$$

which can be estimated using the known points of the track $c(k\Delta s)$, where $k = 0, 1, 2 \ldots$, and Δs is the distance with which the centerline was discretized. Thus, the constrained optimization task in the supervisor is formed through (25), (26) as

$$\underset{\Delta_L}{\operatorname{argmin}} \|u_R + \Delta_L - u_L\|_2^2 \tag{27a}$$

subject to

$$\Delta_L \in [\Delta_{L,min}; \Delta_{L,max}] \tag{27b}$$

$$e_{lat,T} \leq e_{max}. \tag{27c}$$

Infeasibility of (27), e.g., the constraints cannot be guaranteed, is evaluated as an emergency situation for the vehicles. In these cases, the selection of $\Delta_L = 0$ is performed and the maximum braking command to the vehicle is sent.

Although (27) can lead to an accurate solution to the problem, its solution in real-time can lead to difficulties. Therefore, under practical implementations, the continuous solution of (27) can be replaced with the following equal computation process.

1) In most of the operation of the vehicle, it can be considered that the vehicle moves under normal vehicle dynamic conditions, i.e., it is assumed that u_L can result in the keeping of lateral error under e_{max}. Consequently, in the first computation step of the supervisory process, the assumption is checked, such as the ensuring of condition $e_{lat,T} \leq e_{max}$ with u_L.
2) If the assumption in the checking process is validated to be true, an equivalent solution of the optimization process (27) is

$$\Delta_{L,i} = \begin{cases} u_{L,i} - u_{R,i} & \text{if } (u_{L,i} - u_{R,i}) \in [\Delta_{L,min}; \Delta_{L,max}] \\ \Delta_{L,max} & \text{if } (u_{L,i} - u_{R,i}) > \Delta_{L,max} \\ \Delta_{L,min} & \text{if } (u_{L,i} - u_{R,i}) < \Delta_{L,min}, \end{cases} \tag{28}$$

where the indices denote the i^{th} control input. Remark that the selection of the bounds does not necessarily lead to the violation of the lateral error constraint. The achieved lateral error can be influenced by the selection of $\Delta_{L,min}, \Delta_{L,max}$. For example, in the case of low-value selection for the bounds can lead to the limitation of the lateral error.
3) But, if the assumption in the checking process is validated to be false, the optimization process (27) must be performed.

In the case of a practical application, the process above can significantly reduce the computation effort, i.e., the assumption in most of the vehicle operation is verified to be true.

A. Optimization Process for Improving Closed-Loop Performance Level

The formed optimization processes of the robust control design (19), (21) and of the supervisor (27) show that both of them depend on the selection of $\Delta_{L,max}$. In the case of large bound selection, the performance level of the designed robust control can be low, while at low bound selection the impact of u_L on u can be small. Therefore, an optimization process is provided for the selection of $\Delta_{L,max}$, with which the performance level of the control system can be improved, i.e., lap time can be minimized.

The optimization process is based on the evaluation of simulation scenarios. To improve the generality of the achieved solution, each simulation scenario differs from each other. It uses the same simulation environment as for the RL training process, see Section III-B, i.e., the motion of the vehicle model along various tracks is simulated and noise to the lateral position measurement is added. The evaluation is based on the computation of a cost function J, which contains the most important metrics on the

achieved control, such as

$$J = Q_1 \sum_{k=1}^{N} |u_{R,Lat}(k) + \Delta_{L,Lat}(k) - u_{L,Lat}(k)| + Q_2 \sum_{k=1}^{N} |u_{R,Long}(k) + \Delta_{L,Long}(k) - u_{L,Long}(k)| + Q_3(N). \quad (29)$$

In (29) each term contains a metric, which represents the performance of the control system with the given selection of $\Delta_{L,max}$. The terms are computed for each simulation scenario, which has a length with N samples. The value of N is limited to a selected maximum value N_{max}. The first term of (29) represents the impact of $u_{L,Lat}$ on the steering intervention, and the second term reflects the impact of $u_{L,Long}$ on the velocity selection. The priorities of these terms are expressed by scalar weights Q_1, Q_2. Moreover, additional scalar weight Q_3 in the cost is also involved, which reflects the terminal position of the vehicle on the track. Thus, if the vehicle performs the track, i.e., $N < N_{max}$, then $Q_3 = 0$ is selected. But, if the track is not performed, i.e., $N = N_{max}$, then $Q_3 = Q_{3,max}$, $Q_{3,max} > 0$ is selected for penalizing the actual $\Delta_{L,max}$ selection.

During the optimization process the goal is to minimize J, and similarly, to guarantee the robustness of the control, which leads to the minimization process

$$\min_{\Delta_{L,max}} J, \qquad \text{subject to} \qquad 0 < \gamma < 1. \quad (30)$$

Since the candidate values in $\Delta_{L,max}$ can have physical limits, e.g., the achievable steering angle, an efficient solution of the minimization is a line search on a grid of $\Delta_{L,max}$. During the optimization process, a rough grid is defined in the initial step and then, the grid is refined, where the cost has a minimum and the constraint is guaranteed. In (30) the value of γ depends on the $\mathcal{H}_\infty$ controller, and the computation of that is influenced by $\Delta_{L,max}$. Its reason is that Δ_L is part of w, see (17) and thus, the value of $\Delta_{L,max}$ is involved in the $\mathcal{H}_\infty$ control design process as an upper bound. Consequently, in each step of the minimization (30) the $\mathcal{H}_\infty$ controller must be recomputed with the given $\Delta_{L,max}$ of that step. It leads to the γ term of the constraint in (30). Nevertheless, in practice, this process can be simplified if the relationship between $\Delta_{L,max}$ and γ can be precomputed. Consequently, the constraint $0 < \gamma < 1$ can be transformed to the selection range reduction of $\Delta_{L,max}$ if the relationship exists, see the illustration in the next section. The result of the optimization process is $\Delta_{L,max}$, with which the robust control can be designed (21) and the supervisory algorithm (27) can be formed.

VI. Illustration of the Designed Integrated Control System

The goal of this section is to evaluate the performance of the integrated control system through simulation examples. In the examples, the goal of the integrated control system is to minimize the lap time of the vehicle on a given track. The parameters of the vehicle model from the F1TENTH type of indoor test vehicle are derived [43]. The training of the RL-based control agent based on the motion of the vehicle on various tracks has been performed. The neural networks are structured in a way, that it had 3 hidden fully connected layers containing 16, 32, and 64 neurons each and ReLU activation functions. The output layer uses hyperbolic tangent activation to limit the control outputs considering the steering capabilities of the vehicle. For the input measurements of the networks, $N = 5$ is selected with 0.5 m equidistant segments, i.e., 2.5 m horizon ahead of the vehicle is considered. Based on (12), the reward function is defined as $R(s,a) = -0.5x^2_{\text{Lat,err}} - 0.5\delta^2_{ref} + \Delta p$.

The evaluation of the performance level is carried out based on two examinations. First, the optimization process for improving the performance level is illustrated, i.e., the robust control is designed and the value of $\Delta_{L,max}$ is selected. Second, a comparative simulation presents the performance level of the control system, regarding the lap time minimization. All of the presented simulations have been performed on a desktop with an 11th-generation Intel processor.

A. Illustration of Closed-Loop Performance Level Improvement

The optimization process in an example is illustrated, such as the tuning of the robust control and the selection of $\Delta_{L,max}$. The goal of the example is to select $\Delta_{L,max}$ for minimizing cost function (29). Nevertheless, in the current optimization only the bounds on $\Delta_{L,Lat}$ are selected, while the bounds on $\Delta_{L,Long}$ are fixed. Thus, this illustration aims to explore the impact of $\Delta_{L,max,Lat}$ on the performance of the controlled system. The range of $\Delta_{L,Lat}$ is considered to be symmetric, i.e., $\Delta_{L,min,Lat} = -\Delta_{L,max,Lat}$.

In the example the track of each simulation scenario is different, as in the case of the simulation environment for computing (29). The weighting parameters of (29) are selected as $Q_1 = 10; Q_2 = 1; Q_{3,max} = 1000$. The optimization process for finding minimum cost is performed on a grid between $\Delta_{L,max,Lat} = 0.01$ rad. . . 0.5 rad. Around the minimum of the cost a dense grid is selected, and a tighter grid is defined. The results of the optimization process can be found in Fig. 5. It can be seen that the cost value decreases with increasing $\Delta_{L,Lat,max}$, see Fig. 5(a). Its reason is that higher $\Delta_{L,Lat,max}$ provides more possibility for approaching u to u_L. Nevertheless, increasing $\Delta_{L,Lat,max}$ leads to a more reduced level of robustness, i.e., the increasing of γ, see also Fig. 5(a). Since $\gamma < 1$ condition must be guaranteed, $\Delta_{L,Lat,max}$ is increased until this condition is not avoided. The tracking performance of the robust control is illustrated with its maximum lateral error, see Fig. 5(b). This analysis shows that the lateral error using the robust control can be limited to a maximum value. The consequence of the analysis is the selection of $\Delta_{L,Lat,max} = 0.164$ rad, where $\gamma = 0.99$ and $\max(e_{LAT}) = 0.92$ m values are achieved. Thus, in the constraint of the supervisor (27), the upper bound $e_{max} = 0.92$ m is built in, i.e., the performance level of the robust control on the lateral error is considered as the requested minimum performance level against the control system.

The impact of $\Delta_{L,Lat,max}$ on the performance of the control along a curvy section of the exemplary track is also illustrated in Fig. 6. In this case the track is fixed for each scenario to help

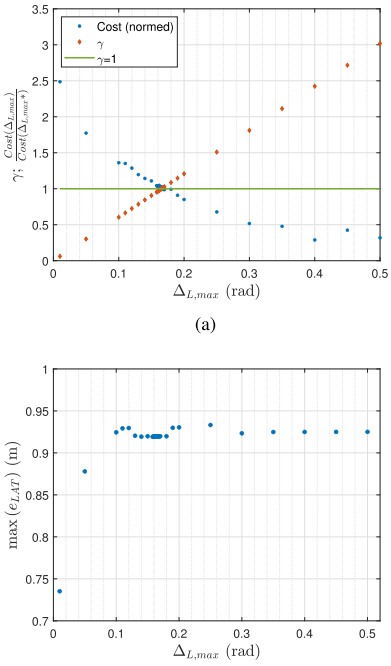


Fig. 5. Analysis of the robust control design. (a) The impact of $\Delta_{L,max}$ on γ, cost (b), and on tracking performance.

the comparison of them. In the case of each scenario, noises are added to the measurements. The results of simulations with some different $\Delta_{L,Lat,max}$ during the reduction of the cost are illustrated. The vehicle path with each $\Delta_{L,Lat,max}$ selection in a curvy section of the circuit is illustrated in Fig. 6(a). Increasing $\Delta_{L,Lat,max}$ leads to the smoothing of the path, i.e., for minimizing lap time. It can also be seen in the improvement of δ signal, see a time section of the simulation in Fig. 6(b). The selection of $\Delta_{L,Lat,max} = 0.01$ leads to fluctuating lateral motion and steering angle, but at higher $\Delta_{L,Lat,max}$ the fluctuation is eliminated. Moreover, the reduction of lap time can be perceived based on the time values of the peaks in the steering signal, e.g., one of the minima of δ is around 30 s using $\Delta_{L,Lat,max} = 0.01$ rad, but the same minimum peak is around 28 s using $\Delta_{L,Lat,max} = 0.164$ rad. Fig. 6(c) shows the achieved lateral error signal, whose characteristics are also smooth at increased $\Delta_{L,Lat,max}$ value and the achievable $\max(e_{LAT})$ is reduced. The velocity profile at each $\Delta_{L,Lat,max}$ selection is found in Fig. 6(d), which shows that the velocity of the vehicle during the selection process is maximized.

The presented illustrations show that the performance level of the control system can be significantly improved through the optimization process. It provides an explainable method for the selection of $\Delta_{L,max}$ from the viewpoint of the robust control design. In the rest of the paper, the designed robust control in the loop loop is built.

B. Illustration of the Closed-Loop Performance Compared to Application of LMPC

The effectiveness of the proposed method through a comparison is illustrated. For this reason, the LMPC [3] with the

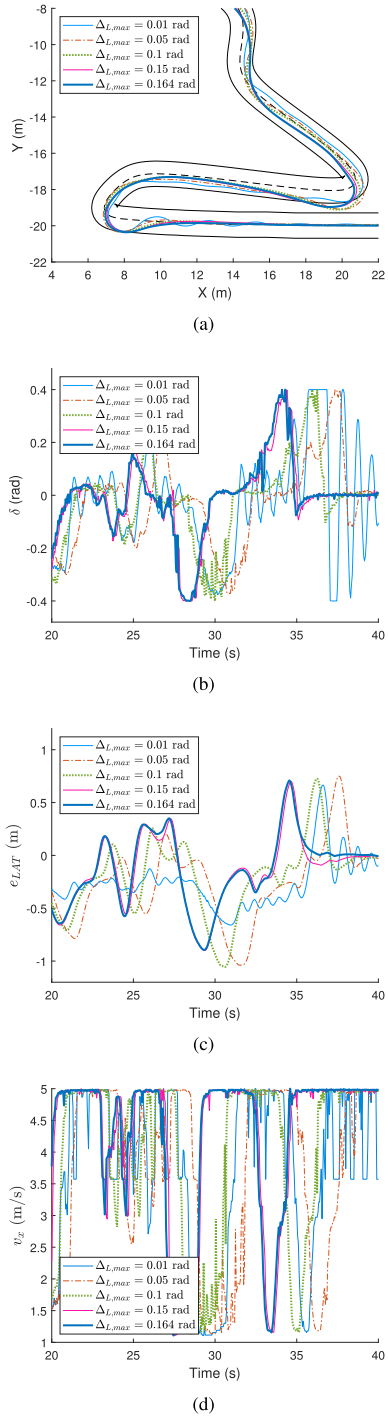


Fig. 6. Impact of $\Delta_{L,max}$ on (a) vehicle trajectory, (b) steering intervention, (c) lateral error, and (d) longitudinal velocity.

available codes (github.com/urosolia/RacingLMPC) has been used. The main objective of the compared methods is to provide vehicle trajectory with minimum lap time. The reward function parameters were chosen to be $A = 0.1$ and $B = 0.5$, to prefer lap time over path tracking but also reduce the unwanted oscillations in the control input. The control inputs of the system are steering angle and longitudinal acceleration. The noises and disturbances in the test environment are greatly reduced compared to what

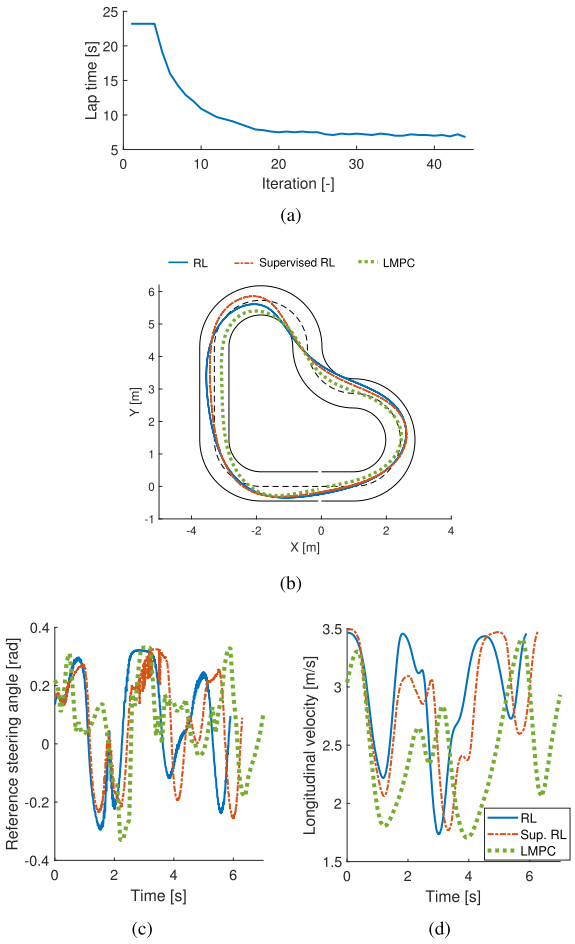


Fig. 7. Convergence plot of the lap time achieved by the LMPC. (a) Comparison of (b) the resulting trajectories, (c) the steering angles and (d) longitudinal velocities in case of the different controllers.

is used in the training of the neural network due to the lack of robustness of the LMPC solution.

The convergence of the lap times of the LMPC can be seen in Fig. 7(a). In the last 20 iterations, the achieved lap time is 6.8 s. The prediction horizon of the LMPC is set to 1.4 s, which results in the length 4.9 m at maximum vehicle velocity (3.5 m/s). The LMPC requires various information on the track to calculate the optimal input signal, e.g., track curvature with higher spatial resolution. During the comparison, all of the other parameters of the vehicle and the environment have been set to be the same in both cases.

Fig. 7(b) illustrates the resulted trajectories on the track, i.e., with LMPC, with (pure) RL, and with the proposed method (Supervised RL). In Fig. 7(c), (d) the operation of the controllers is illustrated by showing the steering interventions and the longitudinal velocity profiles of the vehicles. In the case of pure RL intervention, a more aggressive velocity profile is achieved. In the case of the proposed method, this aggressive intervention is limited by the supervisor, i.e., steering and velocity profiles are closer to the LMPC.

Some numerical results can be found in Table I, such as lap time and average computation time. The average computation time is resulted by the mean of all computation time steps, which

TABLE I
NUMERICAL RESULTS OF THE SIMULATIONS WITH LMPC COMPARISON

Controller	Lap time [s]	Avg. comp. time [ms]
RL	5.86	1.1
Supervised RL	6.29	10.8
LMPC	6.80	21.0

TABLE II
NUMERICAL RESULTS OF THE SIMULATIONS WITH MPCC COMPARISON

Controller	Lap time [s]	Avg. comp. time [ms]
RL	7.12	1.2
Supervised RL	7.76	9.7
MPCC	6.92	91.9

have been performed during each simulation with the trained RL-based controller or the converged LMPC. Additionally, the computation times were in the case of the pure RL in $[0.89,\ 2.27]$ ms, in the case of the Supervised RL in $[4.7,\ 29.0]$ ms and in the case of the LMPC in $[19.7,\ 24.4]$ ms intervals. All simulations have been performed in Python environment. The results show that the using of a pure RL-based controller without the supervisor can lead to reduced lap time and low computation time. Nevertheless, due to safety reasons, the RL-based control with the proposed robust control framework must be used. Thus, the proposed design method can lead to acceptable results, compared to the LMPC method.

C. Illustration of the Closed-Loop Performance Compared to Application of MPCC

In this section the proposed method is compared with an MPCC for Autonomous Racing from [31], which solution aims to minimize lap time. In the case of the MPCC solution, the prediction horizon has been set to 0.8 s and a similar look-ahead distance has been used in the case of the RL-based control agent. The maximal steering angle has been limited to 0.35 rd and the maximum velocity limit is selected to 3.5 m/s in the case of the RL-based control agent. The reward function parameters were chosen to be $A = 0.1$ and $B = 0.1$, to prefer lap time over path tracking and to reduce the oscillations in the steering angle. The reason for B being different is the difference in scale of the vehicle compared to the LMPC example. During the comparison, the same vehicle model is used in the optimization and simulation, which model and the circuit are adapted from the related code (https://github.com/alexliniger/MPCC).

The resulting trajectories, the steering angle signal, and the longitudinal velocities are shown in Fig. 8. It can be seen that each controller can navigate through the track, without leaving it. The MPCC and the RL controller result in slightly different trajectories, especially in the corners with high curvature. Nevertheless, the main objective, i.e., the value of lap time has almost the same performance level, see Table II. The reason for achieving different optima has two reasons. First, the highly nonlinear nature of the vehicle model can lead to a non-convex optimization problem, and thus, different local minima with different trajectories can be achieved. Second, in the case of the two simulations, different longitudinal controls for velocity

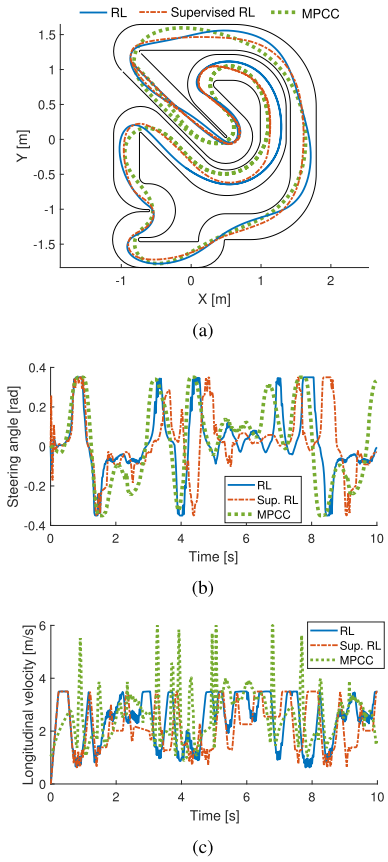


Fig. 8. Comparison of (a) the resulting trajectories, (b) the steering angles, and (c) longitudinal velocities in case of the different controllers.

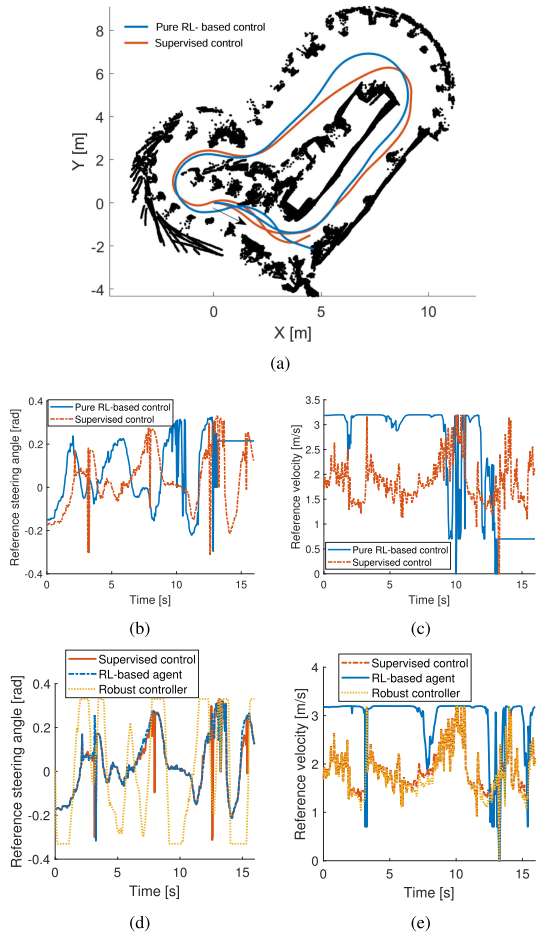


Fig. 9. Estimated trajectories and LiDAR measurements on the track (a). Control inputs in the test scenario: (b) steering angles, (c) reference velocities. Control components of the supervised control intervention at steering angle (d) and reference velocity (e).

tracking are used. Since the longitudinal velocity is limited in the own solution, a slightly larger lap time is resulted.

The largest difference between the two methods is the requested computation time, see Table II, which results correspond to runtime in Matlab environment. The average of the required computation time of one simulation step has been significantly higher in the case of the MPCC. Additionally, the computation times were in the case of the pure RL in $[0.96,\ 2.47]$ ms, in the case of the Supervised RL in $[4.2,\ 25.9]$ ms and in the case of the MPCC in $[57.5,\ 171]$ ms intervals. Its reason is that it solves a complex nonlinear optimization task during the motion of the vehicle, which involves the nonlinear vehicle model. The computation times corresponding to the pure RL controller are significantly lower compared to the other two controllers. Nevertheless, this method can not be used because it is not able to provide a guaranteed minimum performance level in itself.

VII. Demonstration of the Closed-Loop Performance With Implementation

Finally, the operation of the control system is evaluated through real-life experiments using F1TENTH 1:10 small-scaled test vehicle with a two-dimensional LiDAR for environment sensing [44]. The track has been set up using cones with 0.5 m height and environmental objects of the laboratory (e.g., walls). The area of the path covers 10 m $\times$ 12 m and it is a flat polished concrete surface. The shape of the resulting track is independent of the tracks used in the training process.

The test vehicle is localized on the track with a real-time LiDAR-based algorithm. It provides a measured signal on the predicted centerline to the RL-based controller because it requires (x, y) coordinates of the centerline ahead of the vehicle, see (13). The generation process of the centerline contains the following steps. First, the raw LiDAR measurements are converted into a distance image, where every pixel has a value of the distance of the closest measurement point to it. Second, the measured point cloud is filtered to exclude points, which are not resulted by the cones. Third, the numerical gradient magnitude of the image is calculated using the Sobel gradient operator [45]. The centerline is considered to be there, where the gradients have local maxima. Fourth, the centerline is converted into a 1-pixel wide curve using skeletonization [46]. Finally, the resulting pixel coordinates are transformed back to metric coordinates and a spline is fitted into the points to get a continuous line.

A comparison of vehicle trajectories with the supervised and the pure RL-based controls can be seen in Fig. 9(a). In the demonstration scenario the reward function parameters are selected as $A = 0.1$ and $B = 0.5$ to prefer lap time minimization

and to eliminate the oscillations in the steering angle as much as possible without performance degradation. It can be seen that without the supervisor the pure RL-based controller is only able to navigate one full lap and fails on the second by leaving the track. It can be avoided through lateral error prediction of the supervisor, with which the vehicle can complete both laps safely. The control actions in the two cases are shown in Fig. 9(b)–(c). It can be seen that the supervised system selected a reduced velocity profile on the track, resulting in safer maneuvering. In the case of the pure RL-based control, the last turn just before the end of the track was critical, the chosen velocity was high, and the lateral acceleration increased the roll angle significantly resulting in inaccurate LiDAR measurements and an uncertain positioning. The control action signals show significant oscillation around 10 s, which is the consequence of the inappropriate velocity selection. The oscillation in steering intervention resulted in oscillations in the vehicle motion, which led to track departure in the next bend. In the case of the supervised system, the lateral acceleration is limited, i.e., safe motion and trackkeeping were guaranteed. From the viewpoint of lap times, the first lap on the track can be compared. In the case of the pure RL-based control, the achieved minimum lap time is 11.05 s, and in the case of the supervised control, it has 14.24 s value. Although the lap time with the supervisor is increased, the safe vehicle motion during the entire maneuvering can be guaranteed.

Fig. 9(d)–(e) show the operation of the supervised control by comparing the candidate inputs of the RL-based agent and the robust controller. Moreover, the resulting supervised control input is also illustrated. The illustrations show that in the case of lateral control, the actions of the RL-based agent were accepted most of the time by the supervisor, but, in the longitudinal intervention the selection of the safe velocity profile was dominant.

TABLE III
LIST OF SYMBOLS AND NOTATIONS

u	Actual control signal
u_L, u_R	Control signal candidate of the learning-based and robust controller
Δ_L	Output of the supervisor, an addition to the robust control signal
y_L, y_R, y_S	Measurements of the learning-based, robust controller and supervisor
m, Θ_z	Inertial parameter of the vehicle model, mass and moment of inertia
b	Viscous damping
C	Cornering stiffness
α	Lateral slip angle
v_x, v_y	Local velocities
V_x, V_y	Global velocities
ψ	Orientation of the vehicle
δ_{ref}, δ	Reference and actual steering angle
T_δ	Steering dynamics parameter
F_z	Vertical force on the tires
F_{lat}	Lateral tire force
B_t, C_t, D_t, E_t	Pacejka tire characteristic parameters
θ	Actor network parameter vector
π_θ	Policy function (probability distribution)
s_t	Environment sate at time step t
a_t	Action at time step t
τ_t	A trajectory of state and action sequences starting from time step t
γ_{d}	Discount factor in the return calculation
M	Horizon length in the return calculation
R	Reward function
$\mathcal{R}$	Return, the finite-horizon discounted return
V	Value function
A, $\hat{A}$	Advantage and advantage estimate
T	Trajectory length in advantage estimation
r_t	Probability ratio
L	Objective function of RL training
ϵ	Clip parameter of PPO
A, B	Reward function parameters
Δp	Progress along the centerline in the last time step
$x_{\text{lat,err}}$	Lateral path tracking error of the vehicle
x_i, y_i	Local coordinates of points on the path
N	Number of path points in neural network input
$\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$, $\mathcal{D}$,	State space matrices in the robust control design
z_K	Performance vector in robust control design
y_K	Measurement vector in robust control design
$P_{\text{pred},T}$	Predicted vehicle position
$e_{\text{lat},T}$	Predicted lateral error
$e_{\max}$	Maximal allowable lateral error
J	Cost function of Δ_{L} bound optimization
Q_1, Q_2, Q_3,	Cost function parameters

VIII. CONCLUSION

The paper has proposed a robust control design method, which is aided by a learning process. The proposed control strategy has been evaluated via a demonstration on motion optimization. The effectiveness of the method has been presented through comparisons and implementation on a test vehicle. The comparative analysis has shown that the achieved performance level on lap time minimization is close to the LMPC and MPCC-based solutions, but the requested average computation time has been significantly reduced. The results comparisons have illustrated the advantage of modular control structure, i.e., the reduction of computation time is resulted by the separation of the complex control design problem. Moreover, the consequence of the demonstration on a test vehicle is that the proposed control system can be effectively implemented and the achieved results are in line with the simulation-based results. The test scenario has shown that even if the control interventions of the RL agent result in unsafe motion, the supervisory structure can avoid dangerous situations, i.e., leaving the track.

One of the most important future challenges of the proposed method is to enhance the control framework to be able to handle dynamic obstacles on the road, e.g., the motion of pedestrians or other vehicles. It can require the reformulation of the supervisor to consider the stochastic nature of human motion. Moreover, a further challenge is to develop a systematic method to select the bounds in the supervisor, with which method the achieved performance level can be improved. It can require the modification of the control design and the training process, i.e., to find a joint design process for them. Since these control elements have different mathematical structures, their integration on the level of design can pose further theoretical challenges.

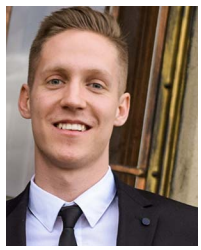
REFERENCES

[1] J. Kabzan, L. Hewing, A. Liniger, and M. N. Zeilinger, "Learning-based model predictive control for autonomous racing," *IEEE Robot. Automat. Lett.*, vol. 4, no. 4, pp. 3363–3370, Oct. 2019.

[2] C. D. McKinnon and A. P. Schoellig, "Learn fast, forget slow: Safe predictive learning control for systems with unknown and changing dynamics performing repetitive tasks," *IEEE Robot. Automat. Lett.*, vol. 4, no. 2, pp. 2180–2187, Apr. 2019.

[3] U. Rosolia and F. Borrelli, "Learning how to autonomously race a car: A predictive control approach," *IEEE Trans. Control Syst. Technol.*, vol. 28, no. 6, pp. 2713–2719, Nov. 2020.

[4] M. Fliess and C. Join, "Machine learning and control engineering: The model-free case," in *Proc. Future Technol. Conf.*, 2021, vol. 1, pp. 258–278.
[5] B. Németh and P. Gáspár, *Guaranteed Performances for Learning-Based Control Systems Using Robust Control Theory*. Cham, Switzerland: Springer, 2021, pp. 109–142.
[6] O. Sename, "Review on LPV approaches for suspension systems," *Electronics*, vol. 10, no. 17, Aug. 2021, Art. no. 2120.
[7] Y. Bao and J. Mohammadpour Velni, "An overview of data-driven modeling and learning-based control design methods for nonlinear systems in LPV framework," in *Proc. Joint 8th IFAC Symp. System Struct. Control, 17th IFAC Workshop Time Delay Syst., 5th IFAC Workshop Linear Parameter Varying Syst.*, 2022, pp. 1–10.
[8] S. Nageshrao et al., *Robust AI Driving Strategy for Autonomous Vehicles*. Cham, Switzerland: Springer, 2023, pp. 161–212.
[9] S. Khosravani, A. Khajepour, B. Fidan, S.-K. Chen, and B. Litkouhi, "Development of a robust vehicle control with driver in the loop," in *Proc. IEEE Amer. Control Conf.*, 2014, pp. 3482–3487.
[10] Y. Chen, X. Zhang, and J. Wang, "Robust vehicle driver assistance control for handover scenarios considering driving performances," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 51, no. 7, pp. 4160–4170, Jul. 2021.
[11] K. Shao, J. Zheng, K. Huang, M. Qiu, and Z. Sun, "Robust model referenced control for vehicle rollover prevention with time-varying speed," *Int. J. Veh. Des.*, vol. 85, no. 1, pp. 48–68, 2021.
[12] S. Br and C. Possieri, "On the use of difference of log-sum-exp neural networks to solve data-driven model predictive control tracking problems," *IEEE Control Syst. Lett.*, vol. 5, no. 4, pp. 1267–1272, Oct. 2021.
[13] W. He, Z. Yan, Y. Sun, Y. Ou, and C. Sun, "Neural-learning-based control for a constrained robotic manipulator with flexible joints," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 12, pp. 5993–6003, Dec. 2018.
[14] M. Zhang, X. Wang, D. Yang, and M. G. Christensen, "Artificial neural network based identification of multi-operating-point impedance model," *IEEE Trans. Power Electron.*, vol. 36, no. 2, pp. 1231–1235, Feb. 2021.
[15] S. Kuutti, R. Bowden, Y. Jin, P. Barber, and S. Fallah, "A survey of deep learning applications to autonomous vehicle control," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 2, pp. 712–733, Feb. 2021.
[16] W. Ruan, X. Huang, and M. Kwiatkowska, "Reachability analysis of deep neural networks with provable guarantees," in *Proc. Int. Joint Conf. Artif. Intell.*, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:19173163
[17] X. Huang, M. Kwiatkowska, S. Wang, and M. Wu, "Safety verification of deep neural networks," in *Proc. 29th Int. Conf. Comput. Aided Verification*, 2017, pp. 3–29.
[18] M. Wu, M. Wicker, W. Ruan, X. Huang, and M. Kwiatkowska, "A game-based approximate verification of deep neural networks with provable guarantees," *Theor. Comput. Sci.*, vol. 807, pp. 298–329, Feb. 2020.
[19] J. O. Pedro, M. Dangor, O. A. Dahunsi, and M. M. Ali, "Dynamic neural network-based feedback linearization control of full-car suspensions using PSO," *Appl. Soft Comput.*, vol. 70, pp. 723–736, 2018.
[20] B. Varga, B. Kulcs1r, and M. H. Chehreghani, "Deep Q-learning: A robust control approach," *Int. J. Robust Nonlinear Control*, vol. 33, no. 1, pp. 526–544, 2023.
[21] B. N9meth and P. G1r, "Ensuring performance requirements for semiactive suspension with nonconventional control systems via robust linear parameter varying framework," *Int. J. Robust Nonlinear Control*, vol. 31, no. 17, pp. 8165–8182, 2021.
[22] Y. Bao and J. M. Velni, "Safe control of nonlinear systems in LPV framework using model-based reinforcement learning," *Int. J. Control*, vol. 96, no. 4, pp. 1079–1090, 2023.
[23] S. D. Kumaravel, A. A. Malikopoulos, and R. Ayyagari, "Optimal coordination of platoons of connected and automated vehicles at signal-free intersections," *IEEE Trans. Intell. Veh.*, vol. 7, no. 2, pp. 186–197, Jun. 2022.
[24] C. Sun, J. Guanetti, F. Borrelli, and S. J. Moura, "Optimal eco-driving control of connected and autonomous vehicles through signalized intersections," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 3759–3773, May 2020.
[25] K. Han, N. Li, E. Tseng, D. Filev, I. Kolmanovsky, and A. Girard, "Improving autonomous vehicle in-traffic safety using learning-based action governor," *Adv. Control Appl.*, vol. 4, no. 2, 2022, Art. no. e101.
[26] L. Shaoshan, *Engineering Autonomous Vehicles and Robots: The DragonFly Modular-Based Approach*. Hoboken, NJ, USA: Wiley, 2020.
[27] P. Karkus, B. Ivanovic, S. Mannor, and M. Pavone, "Diffstack: A differentiable and modular control stack for autonomous vehicles," in *Proc. 6th Annu. Conf. Robot Learn.*, 2023, pp. 2170–2180.
[28] S. Broere, J. Van Kampen, and M. Salazar, "Minimum-lap-time control strategies for all-wheel drive electric race cars via convex optimization," in *Proc. IEEE Eur. Control Conf.*, 2022, pp. 1204–1211.
[29] N. D. Bianco, E. Bertolazzi, F. Biral, and M. Massaro, "Comparison of direct and indirect methods for minimum lap time optimal control problems," *Veh. Syst. Dyn.*, vol. 57, no. 5, pp. 665–696, 2019.
[30] A. Jain and M. Morari, "Computing the racing line using Bayesian optimization," in *Proc. IEEE 59th Conf. Decis. Control*, 2020, pp. 6192–6197.
[31] L. Hewing, A. Liniger, and M. N. Zeilinger, "Cautious NMPC with Gaussian process dynamics for autonomous miniature race cars," in *Proc. IEEE Eur. Control Conf.*, 2018, pp. 1341–1348.
[32] A. Lelkó, B. Németh, D. Fényes, and P. Gáspár, "Integration of robust control with reinforcement learning for safe autonomous vehicle motion," *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 1101–1106, 2023.
[33] A. C. Marosi et al., "Toward reference architectures: A cloud-agnostic data analytics platform empowering autonomous systems," *IEEE Access*, vol. 10, pp. 60658–60673, 2022.
[34] R. Rajamani, *Vehicle Dynamics and Control*. Berlin, Germany: Springer, 2005.
[35] H. B. Pacejka, *Tyre and Vehicle Dynamics*. Oxford, U.K.: Elsevier, 2004.
[36] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," 2018, *arXiv:1506.02438*.
[37] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel, "Trust region policy optimization," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1889–1897.
[38] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.
[39] O. Sename, P. G1r, and J. Bokor, *Robust Control and Linear Parameter Varying Approaches*. Berlin, Germany: Springer, 2013.
[40] C. Scherer and S. Weiland, *Lecture Notes DISC Course on Linear Matrix Inequalities in Control*. Delft, The Netherlands: Delft Univ. Technol., 2000.
[41] S. Boyd, L. E. Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*. Philadelphia, PA, USA: SIAM, 1997.
[42] J. Villagra, B. d'Andréa Novel, M. Fliess, and H. Mounier, "A diagnosis-based approach for tire-road forces and maximum friction estimation," *Control Eng. Pract.*, vol. 19, no. 2, pp. 174–184, 2011.
[43] V. S. Babu and M. Behl, "f1tenth.dev - An Open-source ROS based F1/10 Autonomous Racing Simulator," in *Proc. IEEE 16th Int. Conf. Automat. Sci. Eng.*, 2020, pp. 1614–1620.
[44] M. O'Kelly, H. Zheng, D. Karthik, and R. Mangharam, "F1TENTH: An open-source evaluation environment for continuous control and reinforcement learning," in *Proc. NeurIPS Competition Demonstration Track, Ser. Mach. Learn.*, 2020, vol. 123, pp. 77–89.
[45] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Upper Saddle River, NJ, USA: Prentice-Hall, 2008.
[46] T.-C. Lee, R. L. Kashyap, and C.-N. Chu, "Building skeleton models via 3-D medial surface/axis thinning algorithms," *CVGIP: Graph. Models Image Process.*, vol. 56, no. 6, pp. 462–478, Nov. 1994.



Attila Lelkó received the M.Sc. degree in electrical engineering in 2021 and in mechanical engineering modelling in 2022 from the Budapest University of Technology and Economics, Budapest, Hungary, where he is currently working toward the Ph.D. degree. He is a Young Research Fellow with the Institute for Computer Science and Control, Hungarian Research Network. His research interests include the reliability of neural network-based systems, design of reinforcement learning-based control methods, and the integration of neural networks and model-based control, in relation to autonomous vehicles.

Balázs Németh (Member, IEEE) received the Ph.D. degree in transportation sciences and the habilitation degree from the Budapest University of Technology and Economics, Budapest, Hungary, in 2013 and 2023, respectively. Since 2007, he has been a Senior Research Fellow with the Institute for Computer Science and Control, Hungarian Research Network. His most important research interests include coordinated control design, autonomous vehicle systems, nonlinear analysis, and synthesis of control systems.