Agile Mixed-Integer-based Lane-Change MPC for Collision-Free and Efficient Autonomous Driving

Alexander L. Gratzer, *Student Member, IEEE*, Alexander Schmiedhofer, Alexander Schirrer, and Stefan Jakubek

***Abstract*—Obstacle avoidance and lane-change functionalities are crucial requirements for automated driving that are usually developed to excel in specific road scenarios and therefore, to some extent, lack agility and versatility. This work proposes a versatile vehicle motion controller for connected and autonomous vehicles (CAVs) that, independently of the encountered traffic scenario, realizes safe and efficient lane-change and overtaking maneuvers while implicitly guaranteeing precise obstacle avoidance at all times. This is achieved by introducing a two-layer model predictive control architecture that utilizes mixed-integer-based obstacle avoidance and lane selection formulations. The linear time-invariant optimal control problems are efficiently formulated in Frenet coordinates, incorporate the position predictions of surrounding traffic participants, and guarantee globally optimal solutions. The proposed architecture combines two structurally distinct policies, namely velocity and time-gap tracking, to achieve efficient and safe maneuvering. A method to reduce the computational complexity of the implemented mixed-integer quadratic programming (MIQP) problems is developed. The controller's agility and robustness with respect to multiple complex traffic scenarios are tested in detailed traffic simulations, as well as validated via high-fidelity co-simulations with the CARLA Simulator.**

***Index Terms*—Obstacle avoidance, model predictive control, mixed-integer programming, lane change control, connected and automated driving, single-track model, flatness-based control.**

I. Introduction

LANE changes, usually performed to overtake slower traffic, avoid collisions, or reach a specific lane [1], are among the most complex and dangerous driving maneuvers and pose the main source of (human-caused) traffic accidents on highways [2], [3]. The introduction of connected and automated vehicles (CAVs) aims to eliminate human errors and autonomously perform well-informed, smooth, and collision-free lane changes, which will increase road traffic safety and efficiency [4]–[6]. Connected and automated driving additionally bears the potential to reduce fuel consumption, vehicle emissions, and environmental impact [7]–[9] while increasing the productivity of CAV drivers [10] and the efficiency and comfort of all road users by stabilizing traffic flow and mitigating bottleneck congestion [4]. To realize these benefits, a CAV motion planner should utilize vehicle-to-everything (V2X) communication and collective perception to ensure well-informed maneuvers in diverse and complex road scenarios with a variety of road users [9], [11].

To this end, we propose a holistic model-predictive control (MPC) concept for optimal lane-change planning and vehicle control that implicitly incorporates obstacle avoidance for efficient and collision-free automated driving. The optimal control problem (OCP) is formulated as a mixed-integer quadratic programming (MIQP) problem which considers motion predictions of surrounding traffic participants received via V2X communication. A two-layer MPC architecture realizes efficient lane-change and overtaking maneuvers via a reference velocity tracking policy at the high level while facilitating safe operation via a time-gap tracking policy in the low-level MPC. The versatile control concept is able to efficiently handle and swiftly adapt to a wide range of possible traffic scenarios.

A. Model Predictive Control

Methods suitable for CAV motion planning can be categorized as machine-learning-based, sampling-based, geometry-based, and optimization-based approaches [12], [13]. Being an optimization-based approach, MPC is able to systematically deal with input and state constraints, and utilize behavior predictions of other traffic participants over a receding horizon. The re-planning nature of MPC allows changes in the perceived environment to be accounted for at each evaluation time instance, which ensures robustness against prediction errors and uncertainties [14]. Therefore MPC methods are well-suited for CAV motion planning [15] with efficient obstacle avoidance and lane-change functionality while exploiting V2X communication capabilities [16]. The computational burden for online optimization conducted in MPC algorithms can be high and increases with the complexity of the used prediction models, constraints, and the number of considered obstacles and predicted time steps [11], [17], [18]. Hence, MPC algorithms are usually tailored to specific traffic scenarios or maneuvers to reduce computational effort at the expense of versatility [15], [18]. In contrast to nonlinear MPC (NMPC) problem formulations that are solved via local minimum search algorithms, linear MPC methods guarantee globally optimal solutions [11], [19].

B. Related Work

This section discusses relevant MPC concepts for the efficient planning of lane-change and overtaking maneuvers.

A distributed MPC-based vehicle trajectory planning method implementing a time-varying planning horizon for automated on-ramp highway merging is presented in [20]. Only longitudinal second-order point mass dynamics are considered

Manuscript received 18 March 2024; revised 19 June 2024; revised 11 September 2024; accepted 5 October 2024.

This work was supported by the Austrian Research Promotion Agency (FFG) via the research project *Intelligent Intersection* (ICT of the Future, grant 880830). The authors acknowledge TU Wien Bibliothek for financial support through its Open Access Funding Programme.

The authors are with the Institute of Mechanics and Mechatronics, TU Wien, Vienna, 1060, Austria. E-mail: alexander.gratzer@tuwien.ac.at; alexander.schmiedhofer@tuwien.ac.at; alexander.schirrer@tuwien.ac.at; stefan.jakubek@tuwien.ac.at *(Corresponding author: Alexander L. Gratzer)*

in the linear-quadratic OCP formulation that provides a longitudinal acceleration trajectory. The trajectory planning of CAVs merging from a dedicated CAV lane into an ordinary lane with non-cooperative human-driven vehicles (HDVs) in a two-lane highway scenario is investigated in [21]. The optimal inter-vehicle gap, time, and speed at which a lane change should be initiated are found by utilizing the proposed dynamic programming approach to solve a nonconvex quadratically constrained quadratic programming (QCQP) problem. Again, only longitudinal dynamics are considered and the actual lane-change trajectory is not optimized. A mixed-integer programming (MIP) MPC for longitudinal reference velocity tracking and lane-change decision-making in a two-lane highway scenario excluding lateral control is presented in [22]. Lane selection and longitudinal obstacle avoidance are included in the OCP formulation via binary decision variables preventing inter-vehicle distances from becoming unsafe. While the combinatorial complexity of MIP is mentioned, calculation times are not shown. In [14] the same authors propose a slightly different approach for the MPC-based inter-vehicle space gap and lane-change maneuver starting time for the same two-lane highway road layout. The MIP-based lane selection and obstacle-avoidance formulations used in [22] are substituted with a longitudinal and lateral safety corridor formulation which reduces calculation efforts but also renders the proposed control concept highly specific for the investigated two-lane highway layout. Again, the same two-lane highway road layout is examined in [15]. A longitudinal adaptive model-predictive acceleration controller utilizes the same reference velocity tracking policy and kinematic point-mass model to decide if and when to perform a lane change. A separate weight-adaptive path-tracking MPC controls the steering angle and conducts the lane-change maneuver according to the command of the longitudinal MPC. A nonlinear MPC concept for overtaking unexpected obstacles while maximizing traffic information gain (sensor coverage) in urban road traffic is developed in [23]. Based on the available information, the risk and execution of an overtaking maneuver are decided by a finite state machine. The nonlinear and non-convex OCP is prone to converge to local minima and backup commands are used in case the OCP cannot be solved in time.An algorithm for overtaking a slow-moving leading vehicle in the presence of oncoming and adjacent vehicles with known speeds is proposed in [17]. The OCP is formulated relative to the spatial distance to the leading vehicle and uses ramp barrier constraints for obstacle avoidance in a highway setting. The emanating second-order cone programming (SOCP) problem is solved via SQP which is shown to be computationally more efficient in finding a (locally) optimal solution than the classical MIQP-based temporal OCP formulation. Both OCP formulations show similar results in the investigated simple scenarios under the limiting assumptions of constant leading vehicle speed, zero lateral speed of surrounding vehicles, trivial (point-mass) vehicle dynamics, and a straight two-lane highway layout. The rear and front barrier constraints may not be applicable with more than two lanes and multiple vehicles [24].

Globally optimal autonomous overtaking with consideration of oncoming traffic is investigated in [24]. Obstacle avoidance is realized via the Big-M method which results in an NP-hard MIP MPC problem formulation. Methods to reduce the number of necessary binary variables are proposed to reduce calculation times. However, the method lacks autonomous lane selection capabilities. Additionally, longitudinal and lateral reference trajectories need to be provided externally. An MIQP-based MPC for automated driving that uses a linear vehicle model in a road-aligned coordinate system including obstacle avoidance, lane-change decisions, and traffic rules via mixed-integer inequalities is developed in [25]. The controller shows promising performance in various low-speed ($1\,\mathrm{m/s}$) traffic scenario simulations using the dedicated MIQP solver *BB-ASIPM* [26] but lacks obstacle avoidance and lane-change fidelity, which could render the concept problematic in complex traffic situations, e.g. intersections. A flatness-based linear time-invariant (LTI) obstacle avoidance MPC (OA-MPC) for dynamic CAV motion control in structured and unstructured road environments is proposed in [19]. The concept utilizes an MIQP-based OCP formulation to enable the globally optimal realization of velocity or time gap tracking policies with implicit collision avoidance against dynamic traffic participants. The versatile OA-MPC exploits available position predictions of surrounding vehicles and shows promising performance but lacks real-time computation and lane-change capabilities. The disadvantageous calculation times of the OA-MPC are addressed in [11] by combining it with a computationally fast, real-time capable QP-MPC formulation in a two-layer control architecture, augmenting MIQP-based globally optimal obstacle avoidance with guaranteed locally optimal performance in real time.

C. Research Gap & Contribution

Optimization-based motion planning approaches for CAVs found in most literature are specifically designed for selected road scenarios and tasks, e.g., overtaking, car-following, lane-changing, and merging, in highway, urban, or intersection traffic environments [18]. Often the decision of whether and when to change lanes is separated from the actual motion controller or defined rule-/threshold-based for specific scenarios. While these control concepts excel in their specific area of application, they lack versatility and agility with respect to more generally encountered traffic situations, environments, or constellations. Dynamic and agile (universally applicable) autonomous lane changing and overtaking in variable mixed-traffic environments remains an open research gap [27].

This work extends the discussed OA-MPC formulations [19] and [11] by adding lane-change capabilities, combining velocity tracking and time-gap tracking policies in a two-layer control architecture, and representing the ego vehicle shape in higher fidelity for precise, efficient, and safe autonomous driving. Since real-time computation is already discussed in [11], we do not focus on this aspect here, but present general methods to reduce calculation times of MIQP-based obstacle avoidance problems which can beneficially be incorporated in, e.g., [19] and [11]. We propose to include the selection of the optimal lane directly via binary variables in the cost function of the MIQP-based model-predictive motion controller that, per

design, guarantees collision-free, agile, and globally optimal autonomous driving. The following main contributions are developed hereafter:

1) A model-predictive motion controller for CAVs is proposed which yields globally optimal trajectories by implicitly considering lane selection and the precise avoidance of dynamic obstacles via MIQP. The LC-MPC uses motion predictions of surrounding traffic objects received via V2X communication.
2) The LC-MPC is integrated into a two-layer MPC architecture that combines two structurally distinct policies, namely velocity and time gap tracking, to achieve efficient and safe maneuvering.
3) A method to reduce the calculation times of typically computationally expensive MIQP-based trajectory planning approaches is proposed.
4) The control concept is versatile, robust, and agile with respect to the encountered driving scenarios and traffic participants.
5) A comprehensive validation with a multitude of complex simulation scenarios, including high-fidelity co-simulation studies with the CARLA Simulator [28], is conducted.

The remainder of this work is organized as follows: The problem formulation is given in Sec. II. The LC-MPC formulation, together with a method to reduce its calculation time, is presented in Sec. III and extended into the proposed two-layer LC-MPC architecture in Sec. IV. The performance of the proposed control concept in a multitude of different traffic scenarios is demonstrated in Sec. V and validated via realistic co-simulations in Sec. VI while Sec. VII concludes this paper.

II. Problem Formulation

This work considers the motion planning and control of a single CAV in different road traffic scenarios with an emphasis on efficient lane changes and overtaking under dynamic obstacle avoidance.

A. Control Goals

The following control goals need to be addressed when adding lane-change capabilities to automated driving strategies: (i) Collision safety against static and dynamic obstacles, (ii) stability, and (iii) feasibility with respect to the vehicle dynamics have to be guaranteed at all times. The resulting maneuver should be (iv) efficient while (v) maximizing passenger comfort, and (vi) obeying traffic regulations.

B. Assumptions

This study focuses on lane change maneuver planning and vehicle control with implicit obstacle avoidance, which means that the perception, prediction, and communication modules are assumed to be working satisfactorily. The problem setting is developed based on the following assumptions similar to [14], [18], [25]:

A1) Pre-defined reference paths, usually the center lines of each lane, are available.

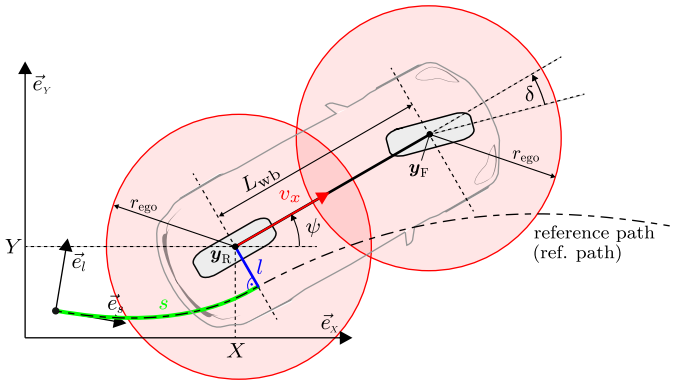

Fig. 1. Kinematic single-track vehicle model incl. ego-shape representation and Frenet coordinates (s, l) with respect to a given reference path, adapted from [11].

A2) An external perception module detects, classifies, and observes relevant traffic participants.
A3) The planar poses and shapes of these detected traffic participants together with their
A4) deterministic motion predictions in the form of position trajectories over a defined prediction horizon are available.
A5) Backward ego vehicle motions are disregarded.

As in [11], the position predictions of CAVs are received via V2X communication while the future motions of HDVs are provided either by V2X communication in combination with intelligent infrastructure and collective perception or an onboard prediction module. We assume the prediction and communication modules are available in line with [14], [18], [25]. We propose a deterministic MPC design, that optionally allows the consideration of uncertain position predictions via adapted obstacle shapes as outlined in Sec. IV-E1.

C. Vehicle Model

The vehicle dynamics used for the control design are modeled according to a kinematic single-track model (non-holonomic, zero slip, also referred to as bicycle model) depicted in Fig. 1. The equations of motion are formulated analogous to [19] as

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{\psi} \\ \dot{v}_x \end{bmatrix} = \begin{bmatrix} v_x \cos\psi \\ v_x \sin\psi \\ v_x \tan(\delta)/L_{\text{wb}} \\ a_x \end{bmatrix}, \tag{1}$$

with the state vector $\boldsymbol{x} = \begin{bmatrix} X, & Y, & \psi, & v_x \end{bmatrix}^{\text{T}}$ consisting of the global Cartesian coordinates X and Y, the global heading angle ψ, and the longitudinal velocity $v_x \geq 0$ (compare assumption A5). Note that here the lateral velocity vanishes, i.e. $v_y \equiv 0$. The input vector $\boldsymbol{u} = \begin{bmatrix} a_x, & \delta \end{bmatrix}^{\text{T}}$ contains the longitudinal acceleration and the steering angle, respectively. The parameter $L_{\text{wb}} > 0$ represents the wheelbase distance. Although (1) by definition does not consider tire slip, the model yields consistent results for limited lateral vehicle accelerations as discussed in [29] and observed in [11] and Sec. VI.

By exploiting the differential flatness property of the kinematic single-track model, (1) can be transformed to

flat coordinates, producing an exactly linearized LTI system comprised of two decoupled double integrators

$$\dot{\boldsymbol{z}} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}}_{\boldsymbol{A}_\mathrm{c}} \boldsymbol{z} + \underbrace{\begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}}_{\boldsymbol{B}_\mathrm{c}} \boldsymbol{\nu} \,, \tag{2}$$

with the flat state vector $\boldsymbol{z}$ and virtual input vector $\boldsymbol{\nu}$ depending on the particular choice of flat outputs $\boldsymbol{y}$. In this work, the Frenet coordinates (s, l), with s being the arc length and l being the lateral deviation with respect to a defined reference path, see Fig. 1, are chosen as flat outputs

$$\boldsymbol{y} = \begin{bmatrix} s, & l \end{bmatrix}^\mathrm{T} \,, \tag{3}$$

which yields the flat state and virtual input vectors

$$\boldsymbol{z} = \begin{bmatrix} s, & \dot{s}, & l, & \dot{l} \end{bmatrix}^\mathrm{T}, \quad \boldsymbol{\nu} = \begin{bmatrix} \ddot{s}, & \ddot{l} \end{bmatrix}^\mathrm{T}. \tag{4}$$

The reference path is assumed to be provided by an environmental perception module (assumption A1) and $v_x = \dot{s}$ and $a_x = \ddot{s}$ hold for nominal motion along it. No analytical transformation between the physical and flat Frenet states is possible for arbitrarily curved reference paths, so the vertices are mapped by utilizing the numerical transformation methods `global2frenet` and `frenet2global` provided by the MATLAB® environment [30]. Representing the vehicle dynamics in (flat) Frenet coordinates allows decoupling of longitudinal and lateral vehicle dynamics control [30], facilitating the implementation of car-following strategies and lateral lane geometry constraints [19]. Limitations of the Frenet frame transformation are discussed in [11]. The LTI vehicle model representation (2) is used in the following for a linear MPC design, which guarantees globally optimal solutions, understood with respect to the transformed (flat) problem description.

III. Generic MPC Formulation

This section presents the LC-MPC formulation for efficient and safe autonomous lane-change control by extending the OA-MPC formulations [19] and [11] with MIQP-based lane selection and more precise obstacle avoidance capabilities. To reduce MIQP calculation times a so-called solution space splitting approach is developed. The resulting linear time-invariant MPC formulation guarantees globally optimal autonomous driving and serves as the basis for the two-layer LC-MPC architecture developed in Sec. IV.

A. Generic Flatness-Based OA-MPC Formulation

The generic OA-MPC formulation used for the LC-MPC design is introduced analogous to [11], [19]. The linearized LTI system dynamics (2) is solved and expressed in discrete time with sampling time T_s under the zero-order hold assumption:

$$\boldsymbol{z}_{k+1} = \underbrace{\begin{bmatrix} 1 & T_\mathrm{s} & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T_\mathrm{s} \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\boldsymbol{A}_\mathrm{d}} \boldsymbol{z}_k + \underbrace{\begin{bmatrix} T_\mathrm{s}^2/2 & 0 \\ T_\mathrm{s} & 0 \\ 0 & T_\mathrm{s}^2/2 \\ 0 & T_\mathrm{s} \end{bmatrix}}_{\boldsymbol{B}_\mathrm{d}} \boldsymbol{\nu}_k \,. \tag{5}$$

It is used as the prediction model for the ego vehicle dynamics. The discrete-time OCP at time step $t_k = k\,T_\mathrm{s}$ with $k \in \mathbb{N}$ is to find the optimal transformed input sequence $\boldsymbol{\mathcal{V}}_k^* = \left[\boldsymbol{\nu}_k^*, \boldsymbol{\nu}_{k+1}^*, .., \boldsymbol{\nu}_{k+N_\mathrm{p}}^*\right]$ and flat state sequence $\boldsymbol{Z}_k^*$ which minimize the convex quadratic objective function (6a) subject to the constraints (6b)–(6d) explained below:

$$\min_{\boldsymbol{\mathcal{V}}, \boldsymbol{Z}, \boldsymbol{s}} \left(J + \boldsymbol{s}^\mathrm{T} r_\mathrm{s}\, \boldsymbol{s} + J_\mathrm{lc} + J_\mathrm{term}\right) \tag{6a}$$

$$\text{s.t. } \boldsymbol{z}_{k+j+1} = \boldsymbol{A}_\mathrm{d}\, \boldsymbol{z}_{k+j} + \boldsymbol{B}_\mathrm{d}\, \boldsymbol{\nu}_{k+j}, \tag{6b}$$

$$\boldsymbol{\nu}_{k+j} \in \mathcal{W}\,, \tag{6c}$$

$$\boldsymbol{z}_{k+j+1} \in \mathcal{Z}\,, \tag{6d}$$

with $j = 0, 1 \ldots, N_\mathrm{p} - 1$. The cost function J is defined in Sec. III-A1, the slack-term J_lc is introduced with the lane-change formulation in Sec. III-C, and optional terminal costs J_term are discussed in [11, Sec. IV-C]. In this work, we omit J_term ($J_\mathrm{term} = 0$) as it is not necessary to obtain a stabilizing closed-loop behavior. The slack cost weight $r_\mathrm{s} \gg 0$ is chosen sufficiently large to enforce collision safety while ensuring problem feasibility. The transformed input set $\mathcal{W}$ is defined in Sec. III-A2, and the flat state set $\mathcal{Z}$, realizing lane keeping (and selection), speed limits, and obstacle avoidance, is discussed in Sec. III-A3 and Sec. III-B.

1) Cost Function: The cost function J is defined as

$$J = \sum_{j=0}^{N_\mathrm{p}-1} \left(\boldsymbol{e}_{k+j+1}^\mathrm{T}\, \boldsymbol{Q}\, \boldsymbol{e}_{k+j+1} + \boldsymbol{\nu}_{k+j}^\mathrm{T} \boldsymbol{R}\, \boldsymbol{\nu}_{k+j}\right), \tag{7}$$

with the tracking error $\boldsymbol{e} = [e_s,\ e_l]^\mathrm{T}$ and tuning matrices $\boldsymbol{Q} = \mathrm{diag}(q_s, q_l)$ and $\boldsymbol{R} = \mathrm{diag}(r_1, r_2)$. The model representation in Frenet coordinates allows the decoupling of lateral and longitudinal control and therefore the essentially independent weighting of the longitudinal resp. lateral cost functionals and virtual inputs (4). The lateral error reads

$$e_{l,k+j+1} := l_{\mathrm{ref},k+j+1} - l_{k+j+1}\,, \tag{8}$$

with $\boldsymbol{l}_{\mathrm{ref},k} = \left[l_{\mathrm{ref},k}, l_{\mathrm{ref},k+1}, .., l_{\mathrm{ref},k+N_\mathrm{p}}\right]$ usually set to zero (compare assumption A1) if lane-change functionalities are disregarded. The longitudinal error is chosen to realize one of two distinct control modes, namely velocity tracking or time-gap tracking:

a) Velocity Tracking: Tracking of a desired reference velocity signal $\boldsymbol{v}_{\mathrm{ref},k} = \left[v_{\mathrm{ref},k}, v_{\mathrm{ref},k+1}, .., v_{\mathrm{ref},k+N_\mathrm{p}}\right]$ is achieved by defining the longitudinal position error as (cf. [11])

$$e_{s,k+j+1} := \left(\sum_{i=0}^{j} T_\mathrm{s}\, v_{\mathrm{ref},k+i}\right) - \left(s_{k+j+1} - s_k\right)\,. \tag{9}$$

On curved roads the lateral vehicle accelerations a_n are limited by reducing $\boldsymbol{v}_{\mathrm{ref},k+i}$ depending on the local reference path curvature $\kappa_\mathrm{ref}(s)$ and utilizing the flat state trajectory of the last time step $\boldsymbol{Z}_{k-1}^*$ (cf. [11])

$$v_{\mathrm{ref},k+i} = \begin{cases} \min\left(v_\mathrm{ref}, \sqrt{\frac{a_{n,\max}}{\left|\kappa_\mathrm{ref}(s_{k-1+i}^*)\right|}}\right) & \text{if } \kappa_\mathrm{ref}(s_{k-1+i}^*) \neq 0 \\ v_\mathrm{ref} & \text{otherwise}\,. \end{cases} \tag{10}$$

As demonstrated in [31], the kinematic single-track model describes vehicle motion accurately up to a maximal lateral acceleration of about $a_{n,\max} \approx 4\,\mathrm{m/s^2}$ on a dry road [11].

b) Time-Gap Tracking: Direct access to s and $\dot{s}$ enables the straightforward implementation of time-gap tracking policies that aim to track an inter-vehicle distance of $h \cdot v$, where h is a chosen fixed time span and v the ego velocity. With $v \approx \dot{s}$ the longitudinal error reads (cf. [11])

$$e_{s,k+j+1} := \underbrace{\sum_{i=0}^{j+1} (h\,\dot{s}_{k+i}) + s_0}_{\text{time-gap}} - \underbrace{\left(s^{\text{pre}}_{k+j+1} - s_{k+j+1} - L_{\text{ego}}\right)}_{\Delta s_{k+j+1}} . \tag{11}$$

Therein, s_0 represent the desired standstill distance and Δs the distance to the predecessor vehicle with L_{ego} being the length of the ego vehicle [16]. The (rear end) position of the predecessor projected on the ego vehicle's reference path is denoted as s^{pre}. The parameters h, s_0 are defined on the tactical level, including the selection of the predecessor vehicle to be followed. One method to realize an overtaking maneuver, for example, is to drop the time-gap tracking objective (11) and switch to the velocity tracking mode (9). Tracking a higher reference velocity than the predecessor together with the realized OA capability automatically induces a safe overtaking maneuver.

2) Input Constraints: The input set in physical coordinates

$$\mathcal{U} = \{\boldsymbol{u} : \boldsymbol{G}_u \boldsymbol{u} \leq \boldsymbol{f}_u\} \tag{12}$$

typically comprises interval constraints on $\boldsymbol{u}$ of the form

$$\mathcal{U} = \{\boldsymbol{u} : \boldsymbol{u}_{\min} \leq \boldsymbol{u} \leq \boldsymbol{u}_{\max}\} . \tag{13}$$

These box constraints are then mapped to the virtual inputs corresponding to the flat coordinates using the state and input trajectories of the last time step's solution $\boldsymbol{X}^*_{k-1}$ and $\boldsymbol{U}^*_{k-1}$, respectively. This allows the formulation of constraints for the transformed inputs $\boldsymbol{\nu}$ according to (cf. [11])

$$\mathcal{W} = \{\boldsymbol{\nu} : \boldsymbol{G}_\nu \boldsymbol{\nu} \leq \boldsymbol{f}_\nu\} . \tag{14}$$

The longitudinal and lateral jerks are limited to $\dot{\nu}_{\max}$ directly in flat coordinates using the virtual inputs computed at the previous time step $\nu_{i,k-1}$ with $i = \{1, 2\}$

$$|\nu_{i,k-1} - \nu_{i,k}| \leq \dot{\nu}_{\max}\, T_{\text{s}} . \tag{15}$$

3) State Constraints: The problem formulation in flat Frenet coordinates and the resulting decoupling of the longitudinal and lateral dynamics allow the direct formulation of state constraints in the flat coordinate space. We formulate the longitudinal velocity constraint as $\dot{s} \leq v_{\max}$ (soft), lane boundary constraints $l_{lhs} \leq l \leq l_{rhs}$ (soft), and $0 \leq \dot{s}$ (hard). Soft constraints are utilized to ensure the solvability of the optimization problem in all cases. Here, soft constraints are formed by re-defining a "hard" inequality constraint of the form $\boldsymbol{g}^{\mathrm{T}}\boldsymbol{u} \leq f$ to $\boldsymbol{g}^{\mathrm{T}}\boldsymbol{u} - s \leq f$, with $s \geq 0$ and high penalty cost on s, in which $\boldsymbol{u}$ and s are decision variables.

B. Obstacle Avoidance Constraints

Obstacle avoidance is realized by avoiding any overlap of the ego vehicle's spatial footprint with any of the modeled *convex obstacle regions* $\mathcal{O}_{i,k}$ at any time step k based on the known or predicted positions, rotations, and suitably inflated shapes of obstacles i. All these quantities are considered known in the scope of this work. We approximate the ego vehicle's shape by two circles with radii r_{ego} centered at the rear $(\boldsymbol{y}_{\text{R}})$ and front axles $(\boldsymbol{y}_{\text{F}})$, compare Fig. 1. Inflating the surrounding obstacle shapes and lane boundaries by r_{ego} allows one to specify the collision avoidance constraints directly with respect to the points $\boldsymbol{y}_{\{\text{R,F}\},k+j} \notin \mathcal{O}_{i,k+j}$. The main difficulty is that these exclusions render the problem landscape non-convex. By utilizing a suitable mixed-integer formulation with auxiliary binary decision variables, a reasonably efficient optimization problem is attained which can be solved to global optimality with modern solver algorithms. MIQP problems are generally NP-hard, so no useful (polynomial) worst-case runtime bounds can be given [32]. However, we present a method to reduce calculation times in Sec. III-D and it becomes evident that on today's hardware, the investigated MIQP-MPC problems can be solved on average in times similar to the required sampling times.

Let a bounded convex polygonal obstacle region $\mathcal{O}$ with n_{e} edges be given in the flat coordinates $\boldsymbol{y}$ as

$$\mathcal{O} = \{\boldsymbol{y} : \boldsymbol{G}\boldsymbol{y} \leq \boldsymbol{f}\} \tag{16}$$

with coefficients $\boldsymbol{G}_{n_{\text{e}}\times 2}$ and $\boldsymbol{f}_{n_{\text{e}}\times 1}$. The well-known Big-M method [33], [34] is utilized to express the exclusion $\boldsymbol{y}_{\{\text{R,F}\}} \notin \mathcal{O}$, by introducing a large constant scalar M (interpreted as a constraint relaxation distance), binary decision variables $\boldsymbol{\delta}_{n_{\text{e}}\times 1} \in \{0,1\}^{n_{\text{e}}}$ and the exclusion constraints (cf. [11])

$$\boldsymbol{f} - \boldsymbol{G}\boldsymbol{y} - s\,\mathbb{1} \leq M\,(\mathbb{1} - \boldsymbol{\delta}) , \tag{17a}$$

$$s \geq 0 , \tag{17b}$$

$$\mathbb{1}^{\mathrm{T}}\,\boldsymbol{\delta} \geq 1 , \tag{17c}$$

which are realized in a soft formulation, with the slack variable $s \in \mathbb{R}$. The binary variables $\boldsymbol{\delta}$ taking value 1 indicate which of the edge constraints in (16) are *violated*, which has to hold true for at least one edge due to (17c) with $\mathbb{1}_{n_{\text{e}}\times 1} = \underbrace{\begin{bmatrix}1, & 1, & \ldots, & 1\end{bmatrix}^{\mathrm{T}}}_{n_{\text{e}}}$. It is evident that when formulating (17) for all known obstacles at all time steps in the problem (6), the globally optimal, feasible and collision-free ego trajectory is obtained if a collision-free solution exists and if assumptions A1-A5 in Sec. II-B hold. Since typically only a few of these constraints are actually relevant, only those OA constraints that are violated otherwise are formulated, and the OCP is re-solved. This iterative sparse constraint formulation leads to significantly faster total MIQP calculation times. A more detailed description of the original OA-MPC formulation can be found in [19] and [11].

We can express $\boldsymbol{y}_{\text{F}} = \begin{bmatrix}s_{\text{F}}, & l_{\text{F}}\end{bmatrix}^{\mathrm{T}}$ in terms of $\boldsymbol{y}\,(= \boldsymbol{y}_{\mathbf{R}})$ and $\boldsymbol{z}$ by

$$\boldsymbol{y}_{\text{F}}(\boldsymbol{z}) = \boldsymbol{y} + \boldsymbol{R}\left(\varphi\left(\dot{s}, \dot{l}\right)\right)\begin{bmatrix}L_{\text{wb}}\\ 0\end{bmatrix} \tag{18}$$

with $\varphi = \arctan\left(\frac{\dot{l}}{\dot{s}}\right)$. Linearizing (18) with respect to $\boldsymbol{Y}_{\mathrm{F},k-1} = \left[\boldsymbol{y}_{\mathrm{F0,k-1}}, \boldsymbol{y}_{\mathrm{F0,k}}, .., \boldsymbol{y}_{\mathrm{F0,k+N_p-1}}\right]$ and $\boldsymbol{Z}_{k-1}$ yields

$$\boldsymbol{y}_{\mathrm{F},k}\left(\boldsymbol{z}_k\right) \approx \boldsymbol{y}_{\mathrm{F0},k} + \underbrace{\frac{\partial\, \boldsymbol{y}_{\mathrm{F}}\left(\boldsymbol{z}\right)}{\partial \boldsymbol{z}}\bigg|_{\boldsymbol{z}_{0,k}}}_{\boldsymbol{L}} \left(\boldsymbol{z}_k - \boldsymbol{z}_{0,k}\right). \tag{19}$$

The exclusion constraints for the center point of the front axle follow by substituting $\boldsymbol{y}$ with (19) into (17a) as

$$\boldsymbol{f} - \boldsymbol{G}\boldsymbol{y}_{\mathrm{F}} - s\,\mathbb{1} \leq M\left(\mathbb{1} - \boldsymbol{\delta}\right) \tag{20a}$$
$$\tilde{\boldsymbol{f}} - \tilde{\boldsymbol{G}}\boldsymbol{z}_k - s\,\mathbb{1} \leq M\left(\mathbb{1} - \boldsymbol{\delta}\right), \tag{20b}$$

with

$$\tilde{\boldsymbol{f}} = \boldsymbol{f} + \boldsymbol{G}\left(\boldsymbol{L}|_{\boldsymbol{z}_{0,k}}\boldsymbol{z}_{0,k} - \boldsymbol{y}_{\mathrm{F0},k}\right), \tag{21a}$$
$$\tilde{\boldsymbol{G}} = \boldsymbol{G}\,\boldsymbol{L}|_{\boldsymbol{z}_{0,k}}. \tag{21b}$$

The exclusion constraints (20b) are concatenated to (17), thus doubling the number of binary decision variables $\boldsymbol{\delta}_{2n_\mathrm{e}\times 1} \in \{0,1\}^{2n_\mathrm{e}}$ and trading increased calculation time for fidelity.

C. Lane Change Constraints

To enable optimal autonomous lane selection, the OCP formulation (6) is adapted by (i) implementing the velocity tracking policy (9), (ii) removing the lateral position tracking from the cost function ($q_l = 0$), and (iii) adding terminal soft constraints on the lateral ego states l and $\dot{l}$ that are switched via binary decision variables c_i and the Big-M method similar to the implementation of the OA constraints in Sec. III-B. The additional terminal soft constraints for a road layout with n_la parallel lanes of width B and the centerline of the middle lane assumed as a global reference path are implemented as

$$\left.\begin{aligned} l_{k+N_\mathrm{p}} &\leq l_{\mathrm{term},i} + s_{\mathrm{lc},1} + M\left(1-c_i\right) \\ l_{k+N_\mathrm{p}} &\geq l_{\mathrm{term},i} - s_{\mathrm{lc},1} - M\left(1-c_i\right) \end{aligned}\right\} l_{k+N_\mathrm{p}} \equiv l_{\mathrm{term},i} \tag{22a}$$

$$\left.\begin{aligned} \dot{l}_{k+N_\mathrm{p}} &\leq 0 + s_{\mathrm{lc},2} \\ \dot{l}_{k+N_\mathrm{p}} &\geq 0 - s_{\mathrm{lc},2} \end{aligned}\right\} \dot{l}_{k+N_\mathrm{p}} \equiv 0 \tag{22b}$$

$$c_i \in \{0,1\} \qquad \sum_{1}^{n_\mathrm{la}} c_i = 1 \qquad l_{\mathrm{term},i} \in \mathcal{L} \tag{22c}$$

with the set of terminal lateral target lane center coordinates $\mathcal{L} = \{0, B, -B, 2B, ..\}$, $|\mathcal{L}| = n_\mathrm{la}$ measured from the global reference path and slack variables $s_{\mathrm{lc},1}$, $s_{\mathrm{lc},2}$ (not to be confused with the Frenet coordinate $y_1 = s$). The cost function (6a) is extended with the slack-term

$$J_\mathrm{lc} = \begin{bmatrix} s_{\mathrm{lc},1}, & s_{\mathrm{lc},2} \end{bmatrix} \begin{bmatrix} r_{\mathrm{lc},1} & 0 \\ 0 & r_{\mathrm{lc},2} \end{bmatrix} \begin{bmatrix} s_{\mathrm{lc},1}, & s_{\mathrm{lc},2} \end{bmatrix}^\mathrm{T}. \tag{23}$$

Since the constraints (22) should just nudge the vehicle to track the center line of a selected lane, the slack-weights $r_{\mathrm{lc},i}$ are chosen suitably low with $r_{\mathrm{lc},i} \ll r_s$. Equations (22) realize a tracking cost function with $n_\mathrm{la}$equivalent global minima. The lane-change extension is applicable to any structured road layout consisting of n_la lanes or reference paths and introduces n_la additional binary decision variables to the original OCP formulation (6). Additionally, individual costs can be assigned to each lane, e.g. to model fast lanes, dedicated lanes for CAVs/HDVs, or right-hand or left-hand driving regulations, by extending (23) to $J_{\mathrm{lc,ext}} = J_\mathrm{lc} + [c_1, .., c_{n_\mathrm{la}}]\,[J_{lc,1}, .., J_{lc,n_\mathrm{la}}]^\mathrm{T}$. The binary decision variables c_i can also be used to implement costs for changing lanes (additionally to the input costs). The exemplary road layout depicted in Fig. 2 with $n_\mathrm{la} = 4$ lanes of width B and the centerline of the second lane used as a global reference path results in the set of terminal lateral target lane center coordinates $\mathcal{L} = \{0, B, -B, 2B\}$ and four additional binary decision variables c_i.

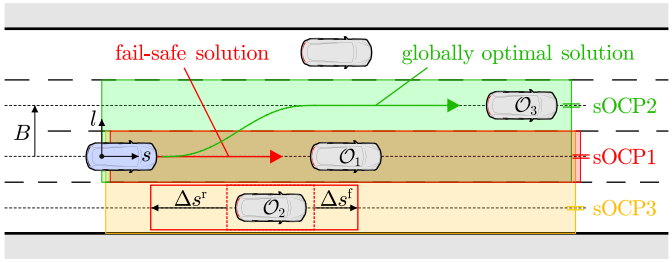

Fig. 2. Splitting the original MIQP problem into smaller sub-problems with visualized sOCP1 - sOCP3 incl. illustration of the original and longitudinally stretched shape of obstacle $\mathcal{O}_2$.

Simulations show that this lane-change extension leads to reduced calculation times and in general to more efficient driving maneuvers by increasing the degrees of freedom of the formulated optimization problem, see also Sec. V.

D. Reducing Calculation Times by Solution Space Splitting

MIP problems are NP-hard [35] and do not provide useful worst-case bounds on solving effort, which is problematic for real-time applications in complex traffic scenarios [19], [36]. Therefore, this section presents a useful method to reduce the solver time of the MIQP-based LC-MPC problem (6). It has to be mentioned that this method still does not deliver bounds on solving effort. This drawback is addressed in [11] by utilizing a two-layer MPC architecture enabling MIQP-based obstacle avoidance in real-time (out of the scope of this work, but exemplarily shown in Sec. V-E).

Domain reduction techniques are known to reduce the calculation times of MIP problems solved by branch-and-bound algorithms drastically [37]. We therefore spatially split the solution space of the original OCP (6) into multiple smaller sub-OCP (sOCP) formulations by shifting the lateral lane boundaries of each sOCP i

$$l_{k+j} \leq l_\mathrm{rhs}^{\mathrm{OCP}i},\; l_{k+j} \geq l_\mathrm{lhs}^{\mathrm{OCP}i},\; \text{with } j = 0,\, 1\,..,\, N_\mathrm{p}-1, \tag{24}$$

and fixing the terminal lane constraints (22a) as depicted in Fig. 2. Here, sOCP1 just considers the current ego vehicle lane, while sOCP2 additionally includes the left and sOCP3 the right lane. To link each sOCP uniquely to its respective target lane (and therefore lane-change maneuver), their terminal target lane constraints (22a) are fixed, e.g., $l_{k+N_\mathrm{p}}^{\mathrm{OCP2}} = B$. We employ a time limit for each sOCPi, $t_{\mathrm{sp,max}}$, and set the cost functions of sOCPs that do not evaluate in that time to $J_i = \infty$. The solution with minimal costs is selected as the control output. If all sOCPs are evaluated in time, we can be sure to have obtained the globally optimal solution to the original problem

formulation. This can be validated by comparing the costs and trajectories of the split and the original OCP formulation.

If legally permitted, overtaking across the oncoming lane is enabled via a dedicated sOCP that includes that domain but uses the original terminal target lane constraints to prevent the ego vehicle from staying in the oncoming lane. To enable this maneuver, the considered prediction horizon needs to be sufficiently large.

1) Obstacle Shape Adaption: In this work, all obstacles are approximated by rectangular shapes ($n_\mathrm{e} = 4$), which, after transformation to flat coordinates, are stretched in s-direction analogous to Fig. 2. The vertices of the rear obstacle edge are shifted by $\Delta s^\mathrm{r} = -h^\mathrm{r}\dot{s}$ while the front edge is shifted by $\Delta s^\mathrm{f} = h^\mathrm{f}\dot{s}^\mathrm{pre}$ with the obstacle's longitudinal speed $\dot{s}^\mathrm{pre}$ and $h^\mathrm{r}, h^\mathrm{f} \in \mathbb{R}_{\geq 0}$. This adaption allows the consideration of speed-dependent longitudinal safety distances and brings the velocity tracking policy closer to a time gap tracking behavior. Setting $h^\mathrm{r} = h^\mathrm{f} = 0$ yields the original definition of $\mathcal{O}$ (16) as proposed in [11], [19] and visualized red-dotted in Fig. 2.

2) Hard Constrained Formulation: The soft-constrained sOCP formulations guarantee problem feasibility but lead to significantly high solver times for non-optimal lane changes. Instead of unnecessarily consuming processing power to find sub-optimal solutions, non-advantageous lane changes should be immediately disregarded. Therefore, we resort to a hard-constrained sOCP formulation, which is faster to solve and, more importantly, does *not* guarantee feasibility. Additionally, we use the resulting terminal longitudinal coordinate of sOCP1 $s_{k+N_\mathrm{p}}^\mathrm{OCP1}$ to implement terminal (hard) constraints for all other sOCPs,

$$s_{k+N_\mathrm{p}} > s_{k+N_\mathrm{p}}^\mathrm{OCP1}\,, \tag{25}$$

to render lane changes (resp. sOCPs) that do not yield a final longitudinal position advantage over the current lane infeasible. Non-relevant lane changes can therefore be quickly filtered out and calculation effort saved. Since a fast and feasible solution of sOCP1 is needed for the implementation of (25), it may be formulated soft-constrained or reformulated as QP. The proposed solution space splitting in combination with hard-constrained sub-OCP formulations and the implementation of the lane-change efficiency constraint (25) reduces solver times by up to $95\,\%$, comp. Fig. 10 in Sec. V-C. Since the sOCPs can be evaluated in parallel, this approach is well-suited for parallel computing and parallel processor applications.

3) Relations to Existing Approaches: There exist approaches that achieve real-time computation by gridding the solution space with a number of trajectories of which the one with the smallest costs is selected as a near optimal solution. In [38] the solution space is discretized by a manifold of candidate trajectories (quintic polynomials) of which the cost functionals are evaluated at each time step. The trajectory with the lowest cost is selected as the "optimal" solution. The solution space splitting presented in this section does not rely on pre-defined trajectories. Each sOCPi provides an optimal trajectory for its defined operation space - if it evaluates in the given time limit. This means that, compared to [38], our approach yields truly globally optimal trajectories that are not constrained by solution space gridding. The individual optimization problems could also be approximated as corridors as in [31], [39] which considers obstacles by time-varying lateral road boundaries and yields faster computation times at the expense of a nonlinear problem formulation without the guarantee of global optimality.

IV. Two-Layer Lane-Change MPC Architecture

Two MPCs are combined in a two-layer control architecture for efficient and collision-free autonomous driving and lane-changing. While the high-level LC-MPC utilizes a velocity-tracking policy for lane selection, the low-level OA-MPC employs a time-gap tracking policy for car-following. Both controllers are based on the generic MPC formulation developed in Sec. III and incorporate obstacle-avoidance capabilities, whereby only the high-level LC-MPC employs lane-selection and solution-space-splitting functionalities.

A. High-Level LC-MPC

The high-level LC-MPC uses the velocity tracking policy (9), incorporates all functionalities presented in Sec. III, and typically employs a larger sampling time $T_\mathrm{s,hi} > T_\mathrm{s,lo}$ than the low-level OA-MPC. The solution space splitting heuristic, cf. Sec. III-D, aims at reduced calculation times and parallel computing. In addition to the hard-constrained domain-restricted sOCPs for lane keeping and lane changing, a soft-constrained OCP formulation for lane-keeping is added. All sOCPs are solved every n-th time step, with $n = {}^{T_\mathrm{s,hi}}/_{T_\mathrm{s,lo}}$. The resulting costs of each sOCP are used as a selection criterion in that the sOCP with the lowest cost is considered optimal and its solution is passed to the low-level MPC. The cost of the previously selected sOCP is discounted by a small factor to introduce some commitment to decided maneuvers and reduce a rapid switching of decisions. Since the soft-constrained sOCP should only be selected if the hard-constrained sOCPs are all (marginally) infeasible or do not evaluate in time, its cost is increased drastically. Compared to its hard-constrained counterpart, the soft-constrained sOCP usually exhibits significantly higher solver times. Therefore, its solution can only be used if the solver time is *fast enough*. Otherwise, a fail-safe solution, which consists of the last optimal solution calculated within the time limit, is implemented. The predicted flat state sequence $\boldsymbol{Z}_k^*$ is passed to the low-level OA-MPC in the form of a reference path (parametrized by $\boldsymbol{z}_1$, $\boldsymbol{z}_3$) together with an optimal speed $\boldsymbol{v}_\mathrm{virt} = \boldsymbol{z}_2$ along it which is used for tracking if no predecessor car or obstacle is detected.

B. Low-Level OA-MPC

The low-level OA-MPC employs the time-gap tracking policy (11) along the reference path provided by the high-level LC-MPC and evaluates over a finer time grid with $T_\mathrm{s,lo} < T_\mathrm{s,hi}$, whereby the same horizon length is chosen for simplicity ($T_\mathrm{s,lo}N_\mathrm{p,lo} = T_\mathrm{s,hi}N_\mathrm{p,hi}$). The LC constraints (22) and obstacle shape adaptions (compare Sec. III-D1) are excluded from the soft-constrained low-level MIQP MPC problem formulation. Time step grouping [40] (clustering of half-space avoidance constraints (16) of adjacent time steps and same orientation and switching each cluster by one respective binary decision variable) is employed to reduce calculation effort.

When following an unobstructed reference path, a virtual vehicle driving with the velocity received from the high-level controller is tracked, comp. Fig. 3 (c). In the vicinity of a relevant predecessor vehicle, the virtual and real predecessor position trajectories are merged, enabling a smooth transition from virtual v_{ref} tracking to actual time gap tracking, see Fig. 4. The tracked time gap is set to be always larger than the stretched obstacle shapes used in the high-level OA-MPC problem formulation, which favors low calculation times of both high- and low-level OCPs. This is due to the following reasons: (i) in nominal operation, the low-level OCP formulation does not require the formulation of OA constraints since it is already provided with a collision-free reference path and velocity, (ii) the low-level OA-MPC keeps enough distance to surrounding vehicles so that the high-level controller does not become marginally infeasible (especially important for hard constrained sOCPs of SSS, see Sec. III-D).

The low-level OA-MPC described here can easily be substituted by the real-time capable two-layer obstacle-avoidance MPC architecture presented in our previous work [11], which would render the TL-LC-MPC architecture real-time capable as exemplarily shown in Sec. V-E. The implemented time gap tracking policy facilitates string stability, the property that state disturbances do not amplify as they propagate along a string of vehicles from any vehicle to its successor [41], [42], and the application of known string stability analysis methods as discussed in [16], [43], [44]. In the case that the low-level controller performs an evasive emergency maneuver due to its obstacle avoidance capabilities (with reduced obstacle shapes), string stable driving is of minor importance. However, in the nominal driving case, the low-level controller is able to provide string-stable operation if properly tuned [16] (out of scope here).

C. Control Architecture

The TL-LC-MPC architecture is illustrated in Fig. 3. The high-level LC-MPC provides a collision-free and (if all OPs evaluate in time) globally optimal solution which is then passed to the low-level OA-MPC as a new reference path and virtual vehicle velocity v_{virt}, see Fig. 3 (b). The low-level OA-MPC tracks a time gap along the received reference path and is able to perform evasive maneuvers utilizing the original obstacle shapes and the whole road as the driving domain, comp. Fig. 3 (c).

D. Feasibility & Stability

The control problems (6) are designed to be always feasible because all state constraints (except the hard constraint ensuring $v_x \geq 0$) are formulated as soft constraints with slacks. Since the SSS method applied in the high-level LC-MPC contains the soft-constrained lane following sub-OCP, at least this solution is always feasible (but most probably not optimal). The feasibility of the hard-constrained control problem can fundamentally be destroyed by critical or malicious obstacles if a collision cannot be avoided by the ego vehicle's control authority. However, in these cases, the proposed soft-constrained problem formulation yields a solution that can be deemed as a sensible trade-off. For example, maximum braking would be employed to minimize collision penalty cost in an unavoidable head-on collision, hence also reducing collision severity in reality.

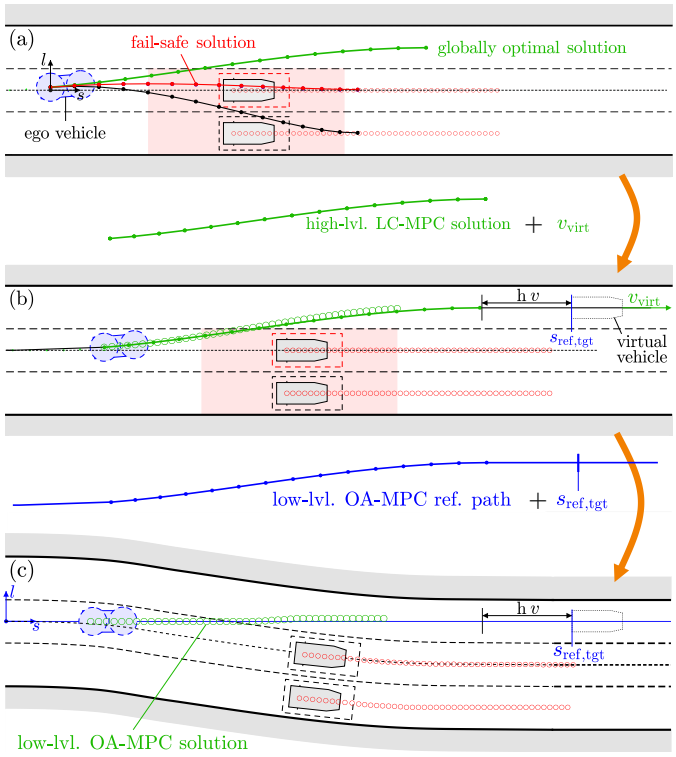

Fig. 3. Illustration of the TL-LC-MPC architecture: The solution of the high-level LC-MPC (a) is handed to the low-level OA-MPC as a reference path and velocity v_{virt} along it (b), which may be used in case no predecessor is in range. The low-level controller (c) tracks a time gap to its predecessor or a virtual vehicle traveling with v_{virt}.

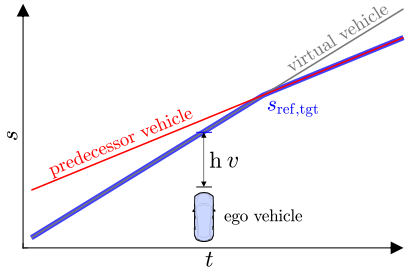

Fig. 4. The longitudinal position reference $s_{\mathrm{ref,tgt}}(t)$ describes the rear end position of the closest preceding vehicle and is tracked in the low-level OA-MPC.

The linear time-invariant core MPC problem can be formulated with the stabilizing Riccati terminal costs $J_{\mathrm{term}} = \boldsymbol{e}_{z,\mathrm{N_p}}^{\mathrm{T}} \boldsymbol{P}\, \boldsymbol{e}_{z,\mathrm{N_p}}$ in (6a) which are formulated in the error coordinates $\boldsymbol{e}_z = \boldsymbol{z} - \boldsymbol{z}^* = \begin{bmatrix} e_s, & \dot{s} - v_{\mathrm{ref}}, & e_l, & \dot{l} \end{bmatrix}^{\mathrm{T}}$. Together with detectability of $(\boldsymbol{A}_{\mathrm{d}}, \boldsymbol{Q})$, stabilizability of $(\boldsymbol{A}_{\mathrm{d}}, \boldsymbol{B}_{\mathrm{d}})$ and a suitable terminal constraint set (not detailed here) nominal closed-loop stability is ensured [45]. We empirically validate the stability and performance of the comprehensive control concept in simulations and co-simulations (i.e., also with realistic model errors) in the following. Therein, none of the mentioned stabilization modifications have been utilized as they were not necessary to obtain stable behavior.

E. Consideration of Uncertain Obstacle Position Predictions

The assumption of having deterministic obstacle predictions available, A4, facilitates straightforward control design. These can be re-interpreted in the case of prediction uncertainties or errors. The deterministic position predictions can be interpreted as the mean loci of (stochastic) occupation probabilities, and the considered obstacle shapes can be inflated in correlation with the variance [11]. Uncertainties in the predictions can be taken into account by increasing the safety margins to the surrounding traffic participants and lane boundaries over the prediction horizon in relation to the confidence level of the assumed prediction modules [11], [14]. E.g., in [46], the probability of a maneuver class and the corresponding probability density function (PDF) of future positions are estimated using Gaussian Mixture regression techniques. The PDF's confidence bounds are used for computing the deterministic obstacle occupancies, which are used in an MPC-based trajectory planner and can be related to the obstacle shapes used in this work. Based on this idea, a method for incorporating uncertain obstacle position predictions by dynamically inflating the corresponding, assumed obstacle shapes is presented in the following.

1) Dynamic Obstacle Shape Inflation: To account for uncertain predictions, extra sOCPs are added in the high-level LC-MPC with dynamically inflated obstacle shapes over the prediction horizon for certain (or all) surrounding vehicles to depict the increasing uncertainties associated with longer predictions, potentially informed by stochastic metrics provided by corresponding estimations, comp. Fig. 5. While lateral inflation accounts for localization errors and uncertain future lane change behavior (and therefore scales with the velocity of the observed vehicle), longitudinal inflation accounts for unforeseen/aggressive acceleration behavior. The costs of each added sOCP are reduced in comparison to the original sOCPs, e.g. in relation to the prediction covariance levels, to strike a balance between efficiency and safety. In line with this approach, assumption A4 can be replaced with the assumption of having access to occupancy probability predictions, comp. [47], and subsequently deriving future mean positions and confidence levels over the prediction horizon. These metrics are then utilized to adjust the obstacle shapes as needed. The impact of uncertain predictions, in conjunction with the proposed obstacle shape adaption method, is investigated in Sec. V-C3, whereas we inflate selected vehicle shapes (i) longitudinally by adding a chosen velocity difference of $10\,\mathrm{km/h}$ to Δs^{f} (which scales with the vehicle velocity), comp. Sec. III-D1, and (ii) laterally from 0 to $2\,B\frac{v}{v_{\mathrm{ref}}}$ over the prediction horizon to account for a possible future lane change. This basic method (which can be easily extended) already allows the consideration of (i) state uncertainties (sensing/perception/localization), (ii) behavioral uncertainties (intention/multi-modal trajectories), and (iii) temporal uncertainties (increase over the length of the prediction horizon).

Uncertain multi-modal driving [48] (beyond a simple lane change) can be considered in the respective sOCPs by the formulation of additional obstacles following each of the possible position trajectories, leading to very conservative driving (not shown here). Further, the control concept is robust against marginal model and obstacle prediction uncertainties via the soft (slack) formulations of the obstacle-avoidance constraints in the low-level OA-MPC [11]. The re-planning nature of receding horizon MPC allows changes in the perceived environment to be accounted for at each time instance, which makes the TL-LC-MPC architecture robust to moderate prediction errors and uncertainties [14].

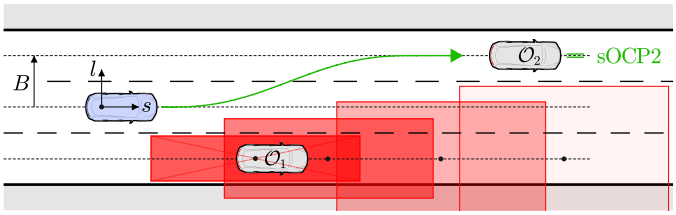

Fig. 5. The ego vehicle keeps a safe distance to $\mathcal{O}_1$ due to its uncertain position predictions (possible future lane change), which are reflected by dynamically inflated obstacle shapes in the lateral direction.

V. Simulation Study

This and the following Sec. VI summarize simulation and co-simulation results of 6 typical traffic scenarios to demonstrate the versatility and agility of the TL-LC-MPC architecture with respect to the encountered traffic situation, environment, and road user composition. The simulation scenarios comprise (A) overtaking with oncoming traffic, (B) dense urban traffic incl. variants (B2) for performance comparison and (B3) considering prediction uncertainties, (C) highway merging, and (D) collision scenario. Furthermore, (E) urban intersection traffic and (F) dense highway traffic are co-simulated with the traffic simulator *CARLA* [28].

A. Simulation Setup

The multi-agent model architecture developed in [49] serves as the MATLAB® simulation environment by representing the traffic participants and their position predictions and providing the necessary information for the optimization problems. The LC-MPC optimization problems (6) are formulated and solved by MIP utilizing the *untuned* commercial solver Gurobi® Optimizer version 11.0.0. The transformation and computation of the MPC control actions are carried out in MATLAB® R2023a. Although parameter tuning could significantly reduce calculation time, especially for MIP, as stated in [50], we use the untuned solver with default settings to allow for easy benchmarking. In the following simulations the TL-LC-MPC architecture-controlled ego vehicle ① is depicted in blue, while all other traffic participants are controlled longitudinally via the IDM car-following model [51] and laterally via the Stanley path-following controller [52], tracking predetermined routes analogous to [49]. All scenarios are conducted with the same set of control parameters listed in Table I, Appendix A. The simulations were carried out on a PC with an Intel Core i9-11900 processor and 64GB RAM.

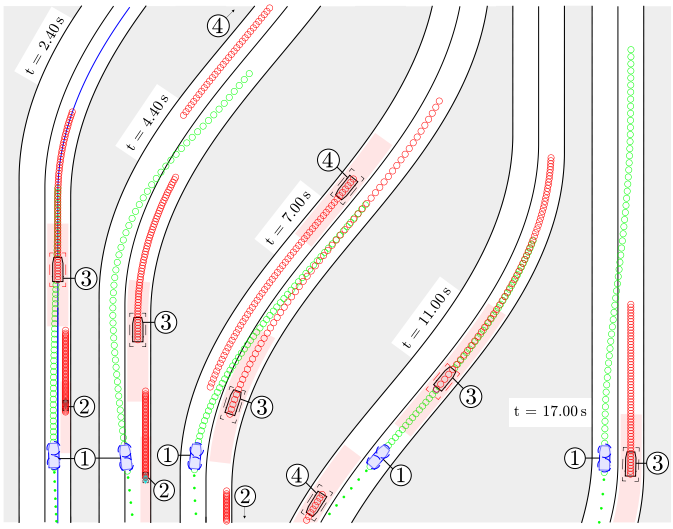

Fig. 6. Scenario (A): The ego vehicle ① takes over a cyclist, aborts an overtaking maneuver due to oncoming traffic and an accelerating vehicle to be overtaken, and finally takes over ③ in the absence of oncoming traffic. At the first time instance the reference path is shown in blue.

B. Urban Traffic Scenario (A)

This scenario shows how the controller deals with curvy road segments, oncoming traffic, and smaller obstacles in the same lane, e.g., bicyclists.

Five selected time instances of the simulation are depicted in Fig. 6 while Fig. 7 shows the corresponding time-series data. The ego vehicle ① spawns with $v_0 = 20\,\mathrm{km/h}$ and accelerates to reach its reference velocity of $v_{\mathrm{ref}} = 50\,\mathrm{km/h}$ while passing a bicyclist ② on its lane and attempting to overtake car ③ that travels with $20\,\mathrm{km/h}$. At $t = 4.5\,\mathrm{s}$ car ③ suddenly accelerates up to $50\,\mathrm{km/h}$ and thereby forces the ego vehicle to abort the overtaking maneuver due to the oncoming traffic ④. After passing ④ and detecting a velocity reduction of ③ back to $20\,\mathrm{km/h}$ the ego vehicle successfully completes the overtaking maneuver. The TL-LC-MPC architecture enables safe and efficient maneuvering and optimally adapts to changing traffic situations. Due to its precise obstacle avoidance capability smaller traffic participants can be passed in the same lane to increase traffic efficiency. Overtaking maneuvers are canceled as soon as the scenario changes adversely and potential collisions are predicted. The efficient OCP formulation in flat Frenet coordinates excels in curved roads and almost always solves in real time, compare Fig. 7. The three low-level solve time peaks at $t = 4.5$ are caused by the changed situation due to the sudden acceleration of vehicle ③. While this causes a peak calculation time of $0.2660\,\mathrm{s}$, the mean calculation time lies with $0.0183\,\mathrm{s}$ still far below $T_{\mathrm{s,lo}}$. The high-level MPC problems are solved on average in $0.0342\,\mathrm{s}$ with a maximum solve time of $0.1330\,\mathrm{s}$.

C. Dense Urban Traffic Scenario (B)

The TL-LC-MPC architecture guides the ego vehicle safely and efficiently through dense urban traffic in real-time, highlighting the advantages of the two-layer architecture in combination with solution space splitting introduced in Sec. III-D.

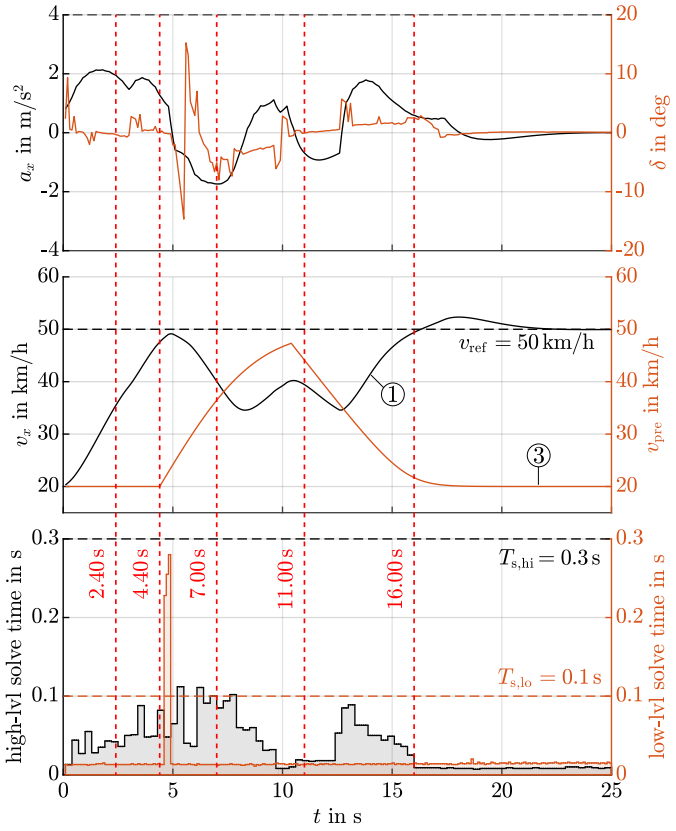

Fig. 7. Scenario (A): Control inputs, velocities of vehicles ① and ③, and high & low-level MPC calculation times incl. snapshot times of Fig. 6.

Fig. 8 depicts five selected time instances of the scenario (B1), while Fig. 9 shows the corresponding time-series data and Fig. 10 displays the different calculation times for each OCP solution space. The ego vehicle ① spawns with $v_0 = 10\,\mathrm{km/h}$ in a group of vehicles ②, ③, ④, and ⑤, whose initial and reference velocities are $10\,\mathrm{km/h}$, $16\,\mathrm{km/h}$, $22\,\mathrm{km/h}$, and $30\,\mathrm{km/h}$, respectively. As the reference speed of the ego vehicle is $v_{\mathrm{ref}} = 30\,\mathrm{km/h}$, it first overtakes the slower vehicles by changing to the center lane and then to the left lane. At $t = 10\,\mathrm{s}$ vehicle ⑤ stops abruptly, forcing ① to move back into the center lane. After passing car ⑤, the ego vehicle can accelerate freely to v_{ref} in the left lane. This scenario shows how efficiently the TL-LC-MPC architecture handles four lane changes during $25\,\mathrm{s}$ of simulation. The computed trajectories are realistic and smooth, while the distance to other vehicles remains within a safe range and the control inputs are far from saturation. Both high- and low-level MPC show real-time capable computation times with a mean & max. solver time of $0.0173\,\mathrm{s}$ & $0.0450\,\mathrm{s}$ and $0.0133\,\mathrm{s}$ & $0.0230\,\mathrm{s}$, respectively, staying well below the controller sampling times of $T_{\mathrm{s,hi}} = 0.3\,\mathrm{s}$ and $T_{\mathrm{s,lo}} = 0.1\,\mathrm{s}$, comp. Fig. 9.

1) Solution Space Splitting: The basic concept of solution space splitting, introduced in Sec. III-D and always applied to the high-level LC-MPC, is visualized in Fig. 10 by showing the calculation times of each sOCP. The selected feasible and optimal sOCP is highlighted in gray for each time step while the red shaded area marks a time step at which all hard-constrained sOCPs turn out to be infeasible. The first three

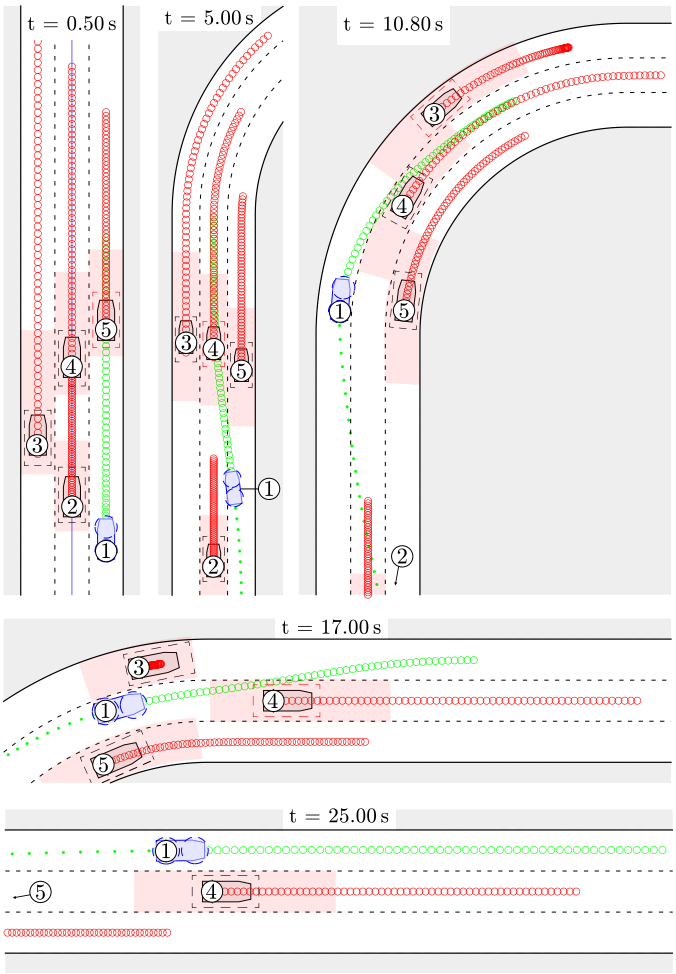

Fig. 8. Scenario (B1): The ego vehicle ① starts on the right lane surrounded by a group of cars with varying velocities. It first changes lanes to the middle and left lane since vehicles ④ and ⑤ move faster. At $t = 10\,\mathrm{s}$ ⑤ breaks which forces the ego vehicle back into the middle lane. After moving past the breaking vehicle ⑤, the ego vehicle chooses a free lane to accelerate to its reference velocity.

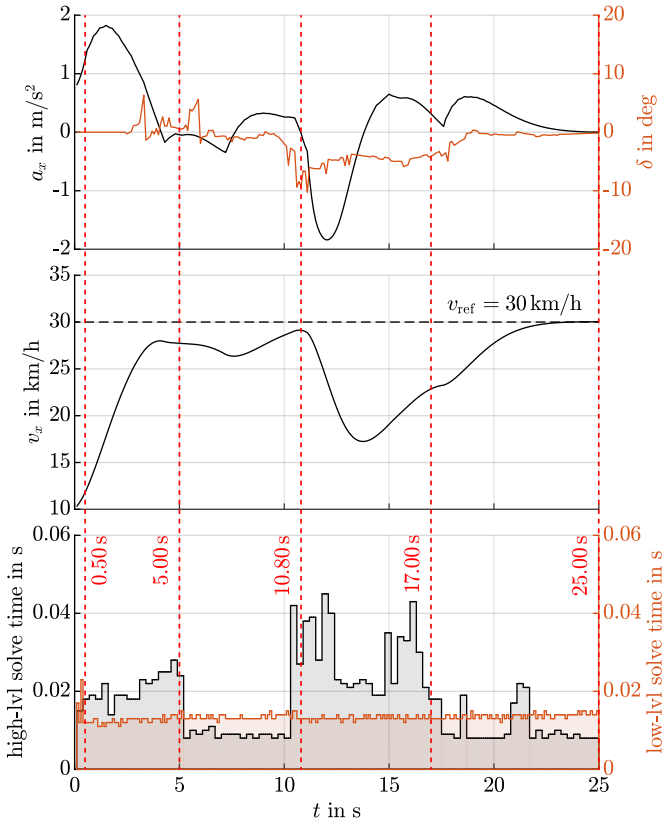

Fig. 9. Scenario (B1): Control inputs, velocities, and high & low-level MPC calculation times incl. snapshot times of Fig. 8.

sOCPs correspond to lane-keeping and lane-changing to the left or right. If the ego vehicle drives in the left or right lane these lane-change sOCPs disappear. The fourth sOCP considers the whole driving domain, therefore its calculation time is typically higher. The soft-constrained sOCP is formulated over the same domain as the first OCP (lane-keeping). The calculation times of the low-level MPC are also displayed in Fig. 10.

2) Comparison to TL-OA-MPC Architecture [11] (B2): Scenario (B) is tested with the ego vehicle controlled by the real-time capable two-layer obstacle avoidance MPC (TL-OA-MPC) architecture recently proposed in [11], which, to create a fair comparison, is enhanced with the lane-change capability developed in Sec. III-C by adding the respective constraints (22a)-(23). Additionally, some parameters are adapted, e.g., the prediction horizon is reduced to 40 samples for a fair comparison (the calculation time of the TL-OA-MPC high-level MIQP controller scales exponentially with it). The TL-OA-MPC controlled ego vehicle stays behind its predecessor longer, abruptly changes to the middle lane, and after overtaking changes back into the right lane. The difference in behavior and calculation time is depicted in Fig. 11. While the position trace of the TL-LC-MPC controlled vehicle corresponds to Fig. 8, the TL-OA-MPC controlled vehicle shows more intense changes in lateral and longitudinal direction. While the calculation times of the low-level controllers are comparable, the high-level calculation times during the overtaking maneuver are an order of magnitude higher, see Fig. 12, because in the TL-OA-MPC architecture, the original MIQP problem is formulated over the entire driving domain.

3) Prediction Uncertainties (B3): We consider the original scenario (B1) with prediction uncertainties to demonstrate the concept of dynamically inflating the obstacle shapes as discussed in Sec. IV-E. Additionally, the ego vehicle uses only a constant velocity prediction to estimate the position trajectories of all other vehicles. The shapes of vehicles ③ and ⑤ are increasingly inflated over the prediction horizon based on their (assumed) prediction covariance levels, see Fig. 13. If possible, the ego vehicle conservatively keeps a safe distance from these vehicles. Here, the obstacle shape inflations also consider multi-modal driving, i.e., a possible future lane change. Compared to the original scenario (B1), the ego vehicle drives more conservatively, which results in $11\,\mathrm{m}$ less distance traveled. The calculation times of the high-level LC-MPC are significantly higher than in scenario (B1) but stay with a mean & max. value of $0.0387\,\mathrm{s}$ & $0.1820\,\mathrm{s}$ well below $T_{\mathrm{s,hi}}$, comp. Fig. 14. The mean low-level OA-MPC solver times increase by ca. $50\,\%$ to $0.0209\,\mathrm{s}$, whereas the max. values do

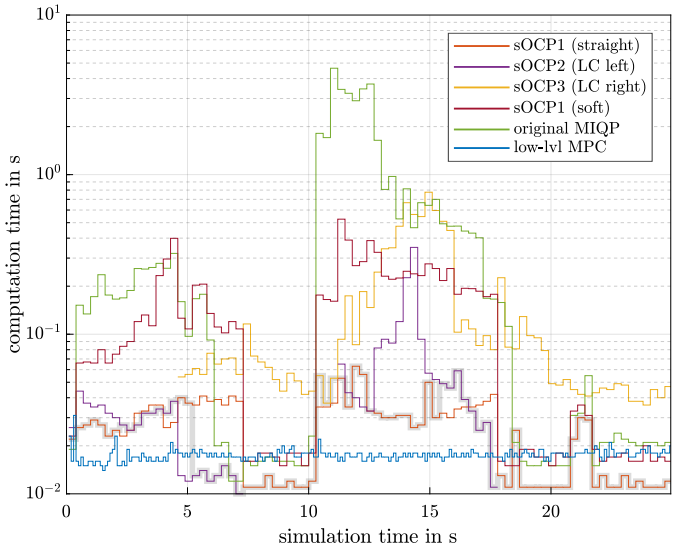

Fig. 10. Scenario (B1): High-level MPC computation times for sOCPs corresponding to different solution spaces. At red-shaded times, all hard constraint OCPs are infeasible. The sOCP whose solution is applied and corresponds to minimal cost is highlighted in gray.

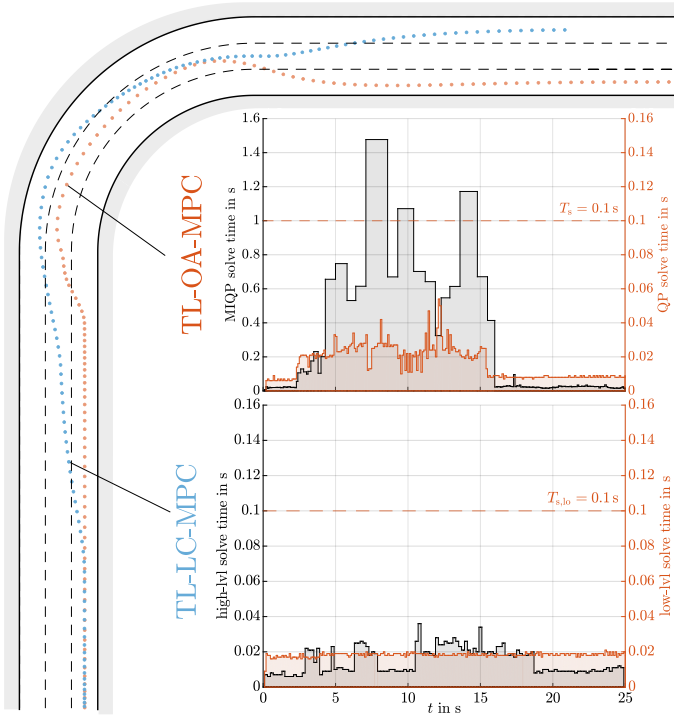

Fig. 11. Scenario (B2): Position traces of ego vehicle controlled by TL-OA-MPC [11] and TL-LC-MPC architecture incl. respective calculation times of high- and low-level MPCs (with different high-level MPC time scales).

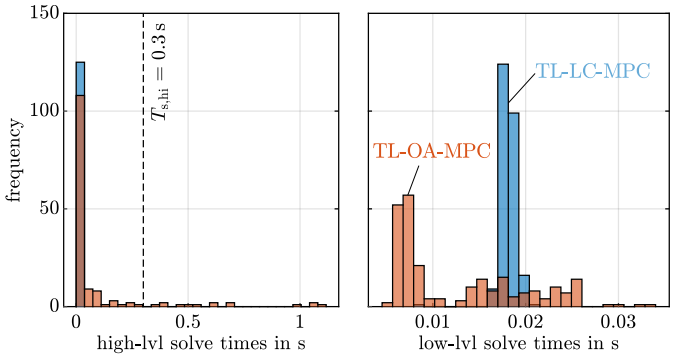

Fig. 12. Scenario (B2): Comparison of high- and low-level MPC solver times of TL-OA-MPC [11] and TL-LC-MPC architectures.

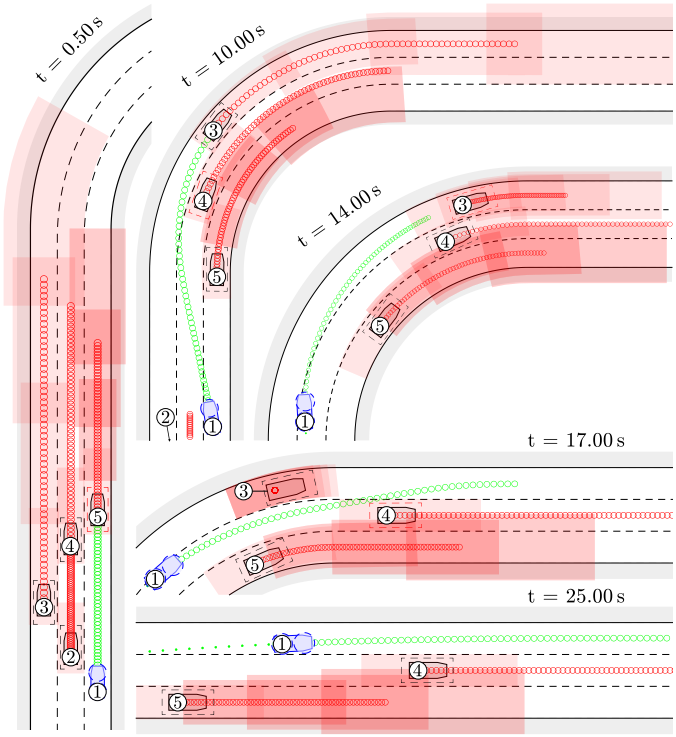

Fig. 13. Scenario (B3): Scenario (B1) with const. velocity-based obstacle position predictions and additionally uncertain position predictions for vehicles ③ and ⑤ considered by the dynamic obstacle shape inflation concept (inflated shapes of ③ and ⑤ visualized for every 5th sample of each position prediction).

not significantly change. Scenario (B3) demonstrates that the TL-LC-MPC architecture is able to deal with uncertain obstacle predictions. For clarity, however, all remaining simulations are carried out following assumption A4.

D. Highway Traffic Scenario (C)

This scenario shows how the controller handles a simple (non-cooperative) highway on-ramp merging scenario performed with a high velocity, and empirically shows string stability (or lane changes that do not alter other vehicles' trajectories). The ego vehicle ① starts in the right merging lane with a velocity of $60\,\mathrm{km/h}$ and has to find a suitable gap to enter the main road while also accelerating to its reference velocity $v_{\mathrm{ref}} = 130\,\mathrm{km/h}$, as shown in the two snapshots in Fig. 15. It accomplishes this by keeping a constant velocity of around $95\,\mathrm{km/hour}$ between $t = 4\,\mathrm{s}$ and $t = 10\,\mathrm{s}$, letting the faster vehicle ② pass, before accelerating and integrating itself into the main traffic in front of vehicle ③ while keeping a safe distance. In this simple scenario, the high- and low-level MPC show real-time capable computation times with mean & max. solver times of $0.0156\,\mathrm{s}$ & $0.0600\,\mathrm{s}$ and $0.0146\,\mathrm{s}$ & $0.0250\,\mathrm{s}$,

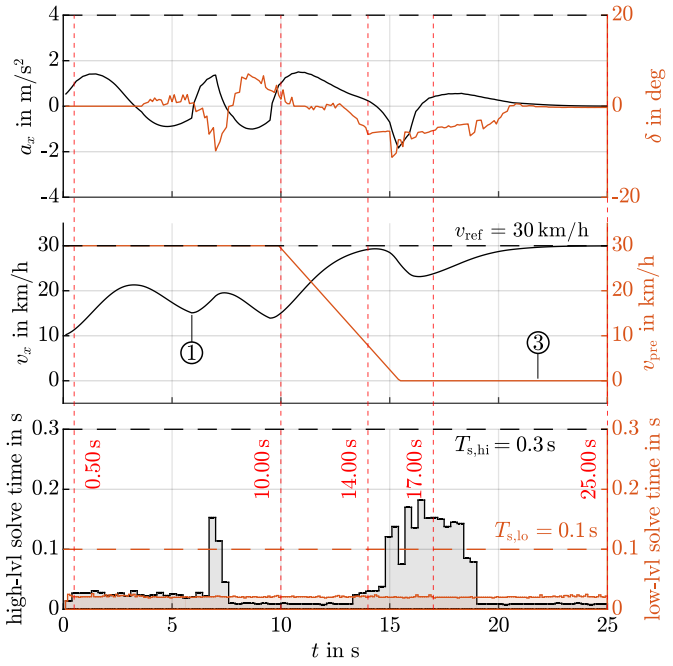

Fig. 14. Scenario (B3): Control inputs, velocities, and high & low-level MPC calculation times incl. snapshot times of Fig. 13.

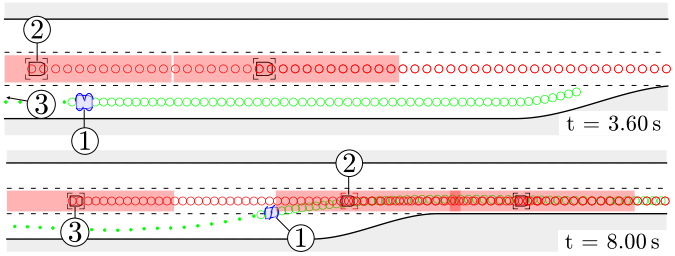

Fig. 15. Scenario (C): The ego vehicle ① has to merge into traffic within the length of the merging lane. It accomplishes that by decelerating until a suitable gap is available. Note that this figure is horizontally compressed.

respectively.

E. Safety-critical Collision Scenario (D)

To demonstrate plausible safety-critical behavior, a scenario involving an unavoidable collision is examined. To provoke a collision, the detection range of the ego vehicle is intentionally reduced to $d_{\mathrm{det}} = 52\,\mathrm{m}$, which is far too short for a reference velocity of $100\,\mathrm{km/h}$. We additionally illustrate a possible coupling of the TL-LC-MPC architecture with the QP-MPC from [11], resulting in a real-time-capable three-layer control architecture.

As soon as the ego vehicle detects the static obstacle at $t = 1.8\,\mathrm{s}$ with an obstacle net distance of $45.3\,\mathrm{m}$, the collision is imminent, comp. Fig. 16. The calculation times of the low-level OA-MPC exceed $T_{\mathrm{s,lo}}$ but the QP-MPC recovers real-time computation and immediately realizes an emergency braking maneuver, see Fig. 17. As a result, the relative speed between the collision participants at the time of collision $t_{\mathrm{coll}} = 4.5\,\mathrm{s}$ is minimized, reducing the kinetic energy at impact by $\Delta E_{\mathrm{k}} \left(v_{\mathrm{ref}}^2 - v_{\mathrm{coll}}^2\right) / v_{\mathrm{ref}}^2 = 95\,\%$. We observe the

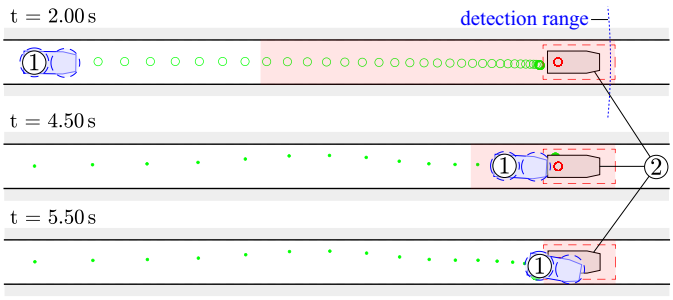

Fig. 16. Scenario (D): The ego vehicle detects the static obstacle ② too late and performs an emergency braking maneuver to reduce the speed at impact. The simulation results are valid until $t_{\mathrm{coll}} = 4.5\,\mathrm{s}$ since collision and post-collision behavior modeling is out of the scope of this work. The overlapping vehicle shapes at $t = 5.5\,\mathrm{s}$ visualize the hypothetical standstill position of the ego vehicle without consideration of interaction with ②.

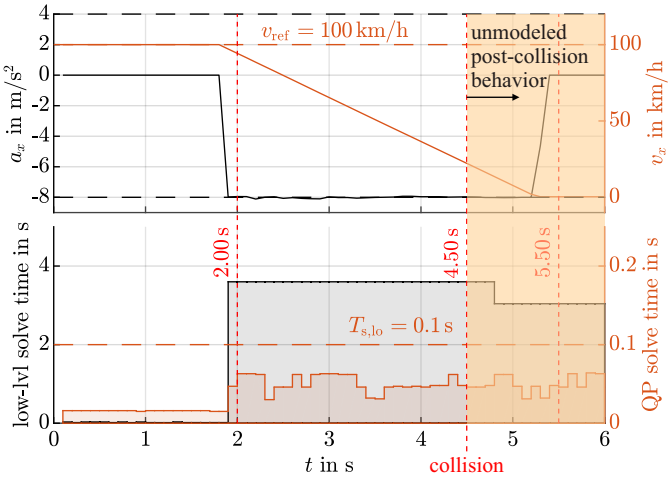

Fig. 17. Scenario (D): Acceleration, velocity, and low-level & QP-MPC calculation times incl. snapshot times of Fig. 16. The ego vehicle collides with the static obstacle ② with a remaining speed of $v_{\mathrm{coll}} = 22.5\,\mathrm{km/h}$.

capability of the control concept to optimally handle this (simple) safety-critical traffic scenario due to the combination with a QP-MPC in a third layer and the slacked obstacle avoidance constraint formulations (17) and (20) guaranteeing real-time computation and problem feasibility. Details regarding the QP-MPC implementation can be found in [11].

VI. Co-Simulation-based Validation

The proposed TL-LC-MPC architecture is validated by realistic co-simulations of highly dynamical scenarios utilizing the traffic simulator for autonomous driving research *CARLA* [28].

A. Co-Simulation Setup

The same co-simulation architecture as in [11] is employed and briefly summarized in this section. For more details see [11], [49]. The vehicle dynamics, approximated for the control design by (5), are computed in CARLA with higher fidelity, including the simulation of (i) longitudinal tire slip, (ii) lateral tire slip, and (iii) drive train dynamics [11], [28]. The robustness of the proposed control architecture with respect to these modeling errors is shown by utilizing the co-simulation architecture

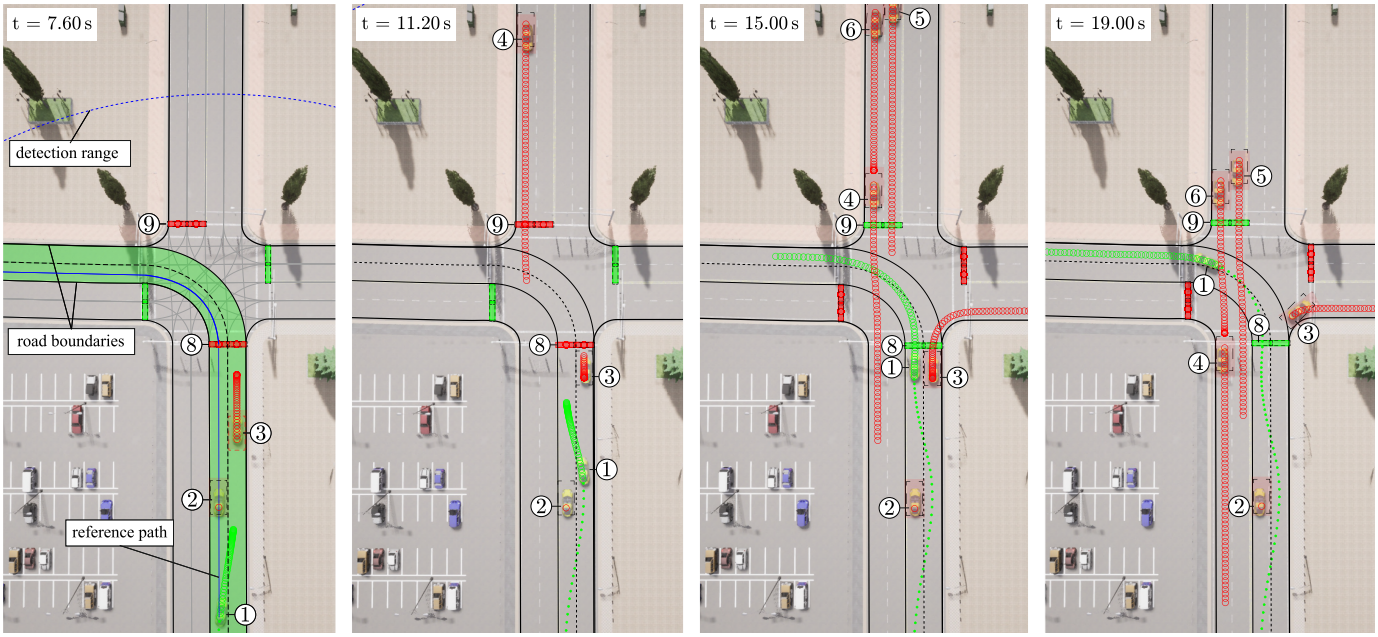

Fig. 18. Scenario (E): Co-simulation of urban intersection: The reference path and road boundary constraints of the ego vehicle ① are highlighted in the first snapshot while the predictions of detected traffic participants $\mathcal{O}_i$ are visualized with red circles. The position prediction and history of the ego vehicle are visualized with green circles and green dots, respectively. Traffic lights are, depending on their phase plan predictions, considered as (static) obstacles.

described in [11]. *CARLA (Car Learning to Act)* is a hyper-realistic (traffic) simulator that uses Unreal Engine 4 to run the simulation and OpenDRIVE standard 1.4 to define roads and urban settings [28], [53]–[56]. Detailed vehicle dynamics are simulated by coupling the components engine, clutch, gears, differential, wheels, tires, suspensions, and chassis. In CARLA vehicles are controlled by the commands of steering, accelerating, and braking, so the calculated acceleration input a of the LC-OA-MPC architecture is mapped to normalized brake and gas pedal positions via a self-developed MATLAB2CARLA bridge that utilizes the provided client API. In this work, CARLA version 0.9.14 is used. *Vehicle control interface*: The control inputs $\boldsymbol{u}_k^*$ are mapped to normalized brake, throttle, and steering inputs via identified look-up tables as an alternative to a low-level PI controller. The Tesla Model 3 vehicle model from the CARLA standard vehicle library is used with adjusted braking torque and engine/drivetrain damping with the clutch engaged, as listed in Table I. More details regarding the employed co-simulation architecture can be found in [11] and [49].

B. Urban Intersection Scenario (E)

In this scenario, the ego vehicle performs an unprotected left turn with oncoming traffic. Selected time instances of the scenario are depicted in Fig. 18 while the ego vehicle states are shown in Fig. 19.

The ego vehicle ① spawns in the left lane with $v_{\mathrm{ref}} = 30\,\mathrm{km/h}$ and employs time gap tracking towards its predecessor ② who has the same velocity. At $t = 6\,\mathrm{s}$, vehicle ② abruptly stops, which causes the ego vehicle to slightly break and change into the right lane. Its new predecessor, vehicle ④, stops at the red traffic light, which triggers a lane change back into the left lane. Although the traffic lights ⑧ and ⑨ are red, all vehicles are assumed to know their phase plans (e.g., via V2X communication) and include this in their OCP formulations. At $t = 15\,\mathrm{s}$ the traffic lights turn green and the ego vehicle continues on its path to turning left. It performs

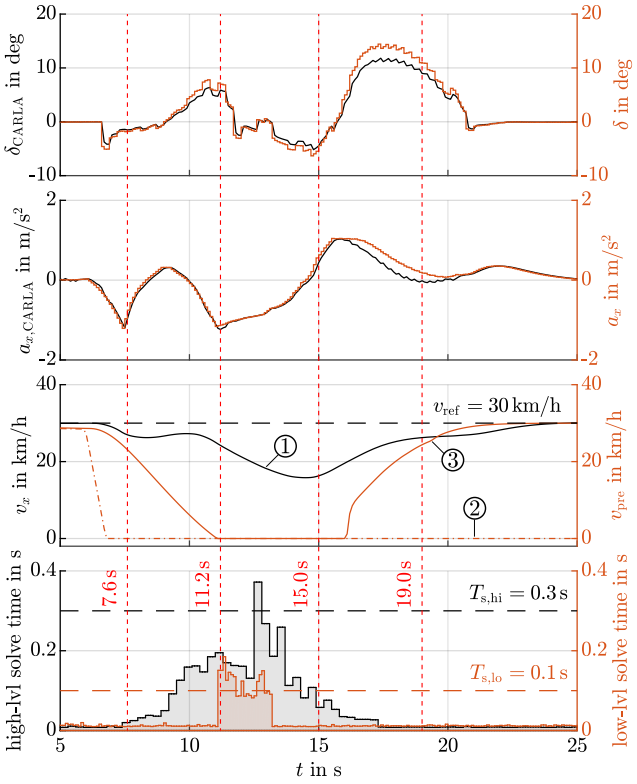

Fig. 19. Scenario (E): Calculated and realized inputs, velocity, and high and low-level calculation times incl. snapshot times of Fig. 18.

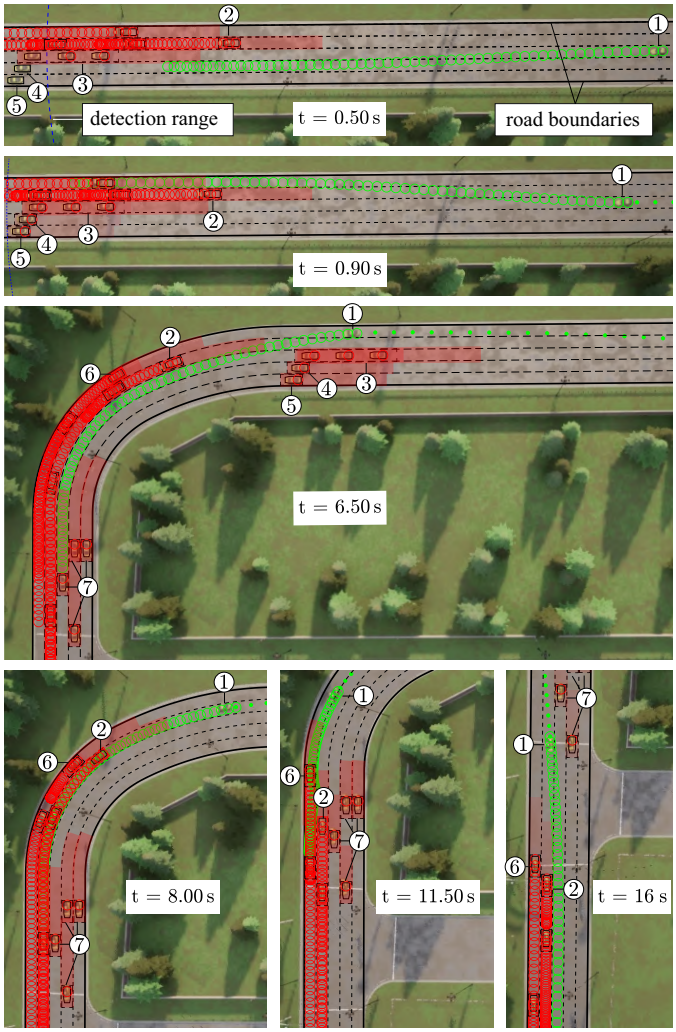

Fig. 20. Scenario (F): Co-simulation of highway: At first, the ego vehicle ① encounters a local optimum at $t = 0.5\,\mathrm{s}$, but with next high-level solution vehicles ④ and ⑤ come into the detection radius and the ego vehicle manages to pass on the right side.

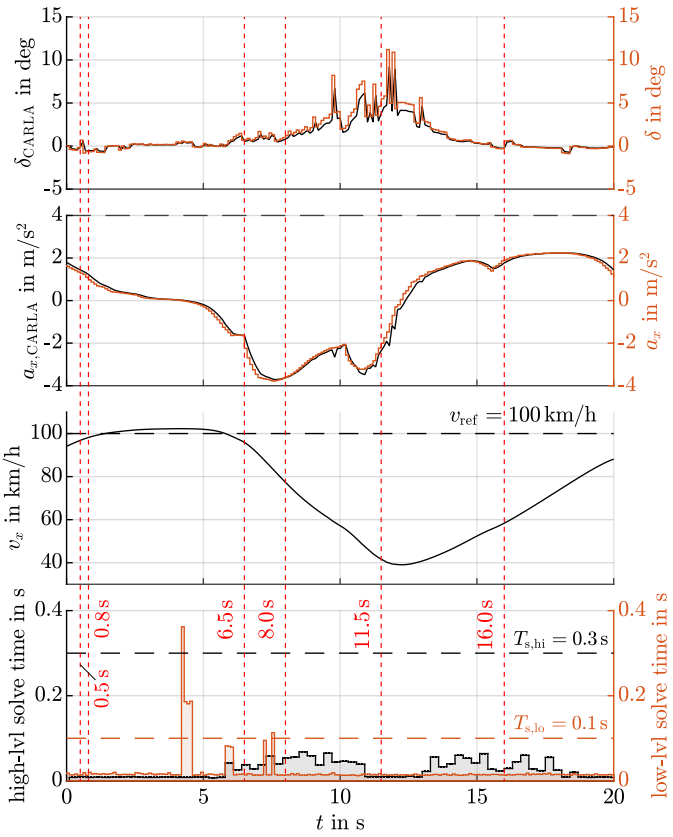

Fig. 21. Scenario (F): Calculated and realized inputs, velocity and high and low-level calculation times incl. snapshot times of Fig. 20.

a final evasive lane change due to the oncoming vehicle ④ to avoid decelerating and finishes the left turn before vehicles ⑤ and ⑥ cross the intersection. The calculation times of the high- and low-level controller are shown in Fig. 19. The high-level MPC solve time only peaks above its sampling time $T_{\mathrm{s,hi}} = 0.3\,\mathrm{s}$ once, with a maximal value of $0.3560\,\mathrm{s}$ while the low-level controller calculation times only peak above $T_{\mathrm{s,lo}} = 0.1\,\mathrm{s}$ between $t = 11\,\mathrm{s}$ and $t = 12\,\mathrm{s}$. The spiking low-level solve times between $t = 11\,\mathrm{s}$ and $t = 13\,\mathrm{s}$ observed in Fig. 19 are caused by active obstacle avoidance conducted in the low-level controller due to OCP solution and sampling time mismatch: While the high-level MPC does not detect a collision with vehicle ④ at $t = 11.1\,\mathrm{s}$, the low-level MPC does one time step later (finer sampling) and applies the corresponding obstacle avoidance constraints, thereby increasing calculation time. The next high-level solution plans to stay in the left lane and brake, but due to a small control output mismatch between the reference velocity tracking and time gap tracking OCP formulations (low-level MPC brakes less intensively) the low-level controller still needs to actively perform obstacle avoidance. Only when the high-level solution plans a lane change to the right lane, at $t = 12.9\,\mathrm{s}$, applied at $t = 13.2\,\mathrm{s}$, the low-level OCP no longer predicts a collision, and thus reduces the calculation times back to nominal levels.

C. Highway Traffic Scenario (F)

This scenario is the same as in [11]. The result of the simulation is depicted at selected times in Fig. 20, while the ego vehicle states are shown in 21. At first, the ego vehicle ① tries to overtake a column of vehicles, ③, on the left side. When vehicles ④ and ⑤ enter the detection radius, it switches to overtake on the right, thereby evading the local optimum created by ③, ④, and ⑤. Between seconds $t = 5\,\mathrm{s}$ and $t = 7\,\mathrm{s}$, the ego vehicle briefly tries to overtake ②, but when it enters the curve it has to reduce its velocity due to the reference velocity shaping, as seen at $t = 8\,\mathrm{s}$. Because of model errors, like tire slip, the ego vehicle drifts onto the most right lane and follows ⑥. When the path is straight again, ① accelerates and overtakes its predecessors ②, ⑥, and the standing group of vehicles ⑦. This scenario is explained in more detail in [11]. As seen in Fig.21, the control inputs stay easily within their limits, the velocity during the whole maneuver is smooth and the calculation times are minimal, spiking only around $t = 5\,\mathrm{s}$ due to obstacle avoidance constraints when the ego vehicle tries to find a gap between ② and ③. Again, the proposed control algorithm shows its effectiveness and versatility in a

complex scenario with many other traffic participants and high velocities.

VII. Conclusion

The proposed TL-LC-MPC architecture allows safe, efficient, and globally optimal autonomous driving for CAVs with computation times close to real-time. Two MIQP-MPCs are coupled, whereby the high-level LC-MPC tracks a reference velocity and decides on optimal lane changes while the low-level controller employs a time-gap tracking policy. A safe time gap is kept to preceding traffic, and optimal lane-change or overtaking maneuvers are executed autonomously whenever beneficial. A method to speed up calculation times of the NP-hard MIQP MPC problems by exploiting parallel computing allows the proposed concept to run in near real-time conditions. Uncertain motion predictions are considered by dynamically inflated obstacle shapes. The control concept is robust and agile with respect to the encountered scenarios, traffic participants, and prediction uncertainties which is demonstrated in numerous different traffic simulations and high-fidelity co-simulation studies conducted with the CARLA Simulator.

The presented approach not only implicitly enables overtaking via its lane-change functionality, but also allows the passing of small obstacles or traffic participants in the same lane if safely possible to increase traffic efficiency and facilitate string stability, which is a future research topic.

Appendix A
Simulation Parameters

The control parameters used in this work are listed in Table I.

TABLE I
Control and simulation parameters

Global Co-Simulation			
$T_{\mathrm{s,cos}}$	0.01	s	sampling time co-sim (CARLA)
T_{s}	0.05	s	samp. time control (non-MPC)
N_{p}	60	samples	prediction horizon
$a_{\max}$	4	$\mathrm{m/s^2}$	maximum acceleration
$a_{\min}$	-8	$\mathrm{m/s^2}$	maximum deceleration
$\delta_{\max}$	± 30	$^\circ$	steering angle saturation
L_{wb}	2.8	m	wheel base
Tesla Model 3 (original $\rightarrow$ adapted value)			
700 $\rightarrow$	2000	N m	maximum brake torque per tire
2 $\rightarrow$	0.15	/	damping rate for zero throttle
TL-LC-MPC Architecture (implemented on vehicle ①)			
$T_{\mathrm{s,hi}}$	0.3	s	high-lvl. MPC sampling time
$T_{\mathrm{s,lo}}$	0.1	s	low-lvl. MPC sampling time
$N_{\mathrm{p,hi}}$	20	samples	high-lvl. MPC pred. horizon
$N_{\mathrm{p,lo}}$	60	samples	low-lvl. MPC pred. horizon
l_{ref}	0	m	lateral offset to reference path
$a_{n,\max}$	4	$\mathrm{m/s^2}$	maximum lateral acceleration
q_s, q_l	$0.25, 0$	/	output weighting (high-lvl.)
q_s, q_l	$0.2, 2$	/	output weighting (low-lvl.)
r_1, r_2	$1, 2$	/	input weighting
r_s	10^6	/	slack weight obst. avoidance
$r_{\mathrm{lc},\{1,2\}}$	10^2	/	slack weight l, $\dot{l}$ constraints
M	100	/	Big M
r_{ego}	1.3	m	ego shape approximation
n_{e}	4	edges	no. of mapped obst. edges
$\mathcal{Q}_{i,\mathrm{infl}}$	6.5×3.1	m	inflated obst. shape measures
h	1.5	s	time gap
s_0	0	m	standstill distance
$\dot{\nu}_{max}$	8	$\mathrm{m/s^3}$	maximal Jerk
d_{det}	100	m	detection radius in (B), (E)
	52	m	detection radius in (D)
	180	m	detection radius in (F)
	250	m	detection radius in (A), (C)
v_{ref}	30	km/h	reference velocity in (B), (E)
	50	km/h	reference velocity in (A)
	100	km/h	reference velocity in (D), (F)
	130	km/h	reference velocity in (C)

References

[1] P. Chauhan, V. Kanagaraj, and G. Asaithambi, "Understanding the mechanism of lane changing process and dynamics using microscopic traffic data," *Physica A: Stat. Mech. its Appl.*, vol. 593, p. 126981, 2022.

[2] T. Peng, L. Su, R. Zhang, Z. Guan, H. Zhao, Z. Qiu, C. Zong, and H. Xu, "A new safe lane-change trajectory model and collision avoidance control method for automatic driving vehicles," *Expert Syst. with Appl.*, vol. 141, p. 112953, 3 2020.

[3] A. Pande and M. Abdel-Aty, "Assessment of freeway traffic parameters leading to lane-change related collisions," *Accident Anal. Prev.*, vol. 38, no. 5, pp. 936–948, 2006.

[4] Y. Wang, L. Wang, J. Guo, I. Papamichail, M. Papageorgiou, F. Y. Wang, R. Bertini, W. Hua, and Q. Yang, "Ego-efficient lane changes of connected and automated vehicles with impacts on traffic flow," *Transp. Research Part C: Emerg. Technol.*, vol. 138, no. April, p. 103478, 2022.

[5] S. E. Shladover, "Connected and automated vehicle systems: Introduction and overview," *J. Intell. Transp. Syst. Technol. Planning, Oper.*, vol. 22, no. 3, pp. 190–200, 2018.

[6] Z. Nie and H. Farzaneh, "Energy-efficient lane-change motion planning for personalized autonomous driving," *Appl. Energy*, vol. 338, no. January, p. 120926, 5 2023.

[7] J. Guo, U. Kurup, and M. Shah, "Is it Safe to Drive? An Overview of Factors, Metrics, and Datasets for Driveability Assessment in Autonomous Driving," *IEEE Trans. on Intell. Transp. Syst.*, vol. 21, no. 8, pp. 3135–3151, 2020.

[8] L. Claussmann, M. Revilloud, D. Gruyer, and S. Glaser, "A Review of Motion Planning for Highway Autonomous Driving," *IEEE Trans. on Intell. Transp. Syst.*, vol. 21, no. 5, pp. 1826–1848, 5 2020.

[9] D. Cao, X. Wang, L. Li, C. Lv, X. Na, Y. Xing, X. Li, Y. Li, Y. Chen, and F. Y. Wang, "Future Directions of Intelligent Vehicles: Potentials, Possibilities, and Perspectives," *IEEE Trans. on Intell. Veh.*, vol. 7, no. 1, pp. 7–10, 2022.

[10] W. Schwarting, J. Alonso-Mora, and D. Rus, "Planning and Decision-Making for Autonomous Vehicles," *Annu. Review Control. Robotics, Auton. Syst.*, vol. 1, pp. 187–210, 2018.

[11] A. L. Gratzer, M. M. Broger, A. Schirrer, and S. Jakubek, "Two-Layer MPC Architecture for Efficient Mixed-Integer-Informed Obstacle Avoidance in Real-Time," *IEEE Trans. on Intell. Transp. Syst.*, pp. 1–18, 2024.

[12] Z. Han, Y. Wu, T. Li, L. Zhang, L. Pei, L. Xu, C. Li, C. Ma, C. Xu, S. Shen, and F. Gao, "An Efficient Spatial-Temporal Trajectory Planner for Autonomous Vehicles in Unstructured Environments," *IEEE Trans. on Intell. Transp. Syst.*, pp. 1–18, 4 2023.

[13] B. Li, Y. Zhang, Y. Ouyang, Y. Liu, X. Zhong, H. Cen, and Q. Kong, "Fast Trajectory Planning for AGV in the Presence of Moving Obstacles: A Combination of 3-dim A Search and QCQP," in *2021 33rd Chinese Control and Decision Conference (CCDC)*. IEEE, 5 2021, pp. 7549–7554.

[14] J. Nilsson, M. Brannstrom, E. Coelingh, and J. Fredriksson, "Lane Change Maneuvers for Automated Vehicles," *IEEE Trans. on Intell. Transp. Syst.*, vol. 18, no. 5, pp. 1087–1096, 2017.

[15] Y. Liang, Y. Li, A. Khajepour, Y. Huang, Y. Qin, and L. Zheng, "A Novel Combined Decision and Control Scheme for Autonomous Vehicle in Structured Road Based on Adaptive Model Predictive Control," *IEEE Trans. on Intell. Transp. Syst.*, vol. 23, no. 9, pp. 16 083–16 097, 2022.

[16] A. L. Gratzer, S. Thormann, A. Schirrer, and S. Jakubek, "String Stable and Collision-Safe Model Predictive Platoon Control," *IEEE Trans. on Intell. Transp. Syst.*, vol. 23, no. 10, pp. 19 358–19 373, 10 2022.

[17] J. Karlsson, N. Murgovski, and J. Sjoberg, "Computationally Efficient Autonomous Overtaking on Highways," *IEEE Trans. on Intell. Transp. Syst.*, vol. 21, no. 8, pp. 3169–3183, 2020.

[18] J. Nilsson, M. Brännström, J. Fredriksson, and E. Coelingh, "Longitudinal and Lateral Control for Automated Yielding Maneuvers," *IEEE Trans. on Intell. Transp. Syst.*, vol. 17, no. 5, pp. 1404–1414, 2016.

[19] A. L. Gratzer, M. M. Broger, A. Schirrer, and S. Jakubek, "Flatness-Based Mixed-Integer Obstacle Avoidance MPC for Collision-Safe Automated Urban Driving," in *9th International Conference on Control, Decision and Information Technologies (CoDIT)*. Rome: IEEE, 7 2023, pp. 1844–1849.

[20] Y. Zhou, M. E. Cholette, A. Bhaskar, and E. Chung, "Optimal Vehicle Trajectory Planning with Control Constraints and Recursive Implementation for Automated On-Ramp Merging," *IEEE Trans. on Intell. Transp. Syst.*, vol. 20, no. 9, pp. 3409–3420, 9 2019.

[21] C. Yang, X. Chen, X. Lin, and M. Li, "Coordinated trajectory planning for lane-changing in the weaving areas of dedicated lanes for connected and automated vehicles," *Transp. Research Part C: Emerg. Technol.*, vol. 144, 11 2022.

[22] J. Nilsson and J. Sjoberg, "Strategic decision making for automated driving on two-lane, one way roads using model predictive control," in *IEEE Intelligent Vehicles Symposium, Proceedings*, 2013, pp. 1253–1258.

[23] H. Andersen, J. Alonso-Mora, Y. H. Eng, D. Rus, and M. H. Ang, "Trajectory Optimization and Situational Analysis Framework for Autonomous Overtaking With Visibility Maximization," *IEEE Trans. on Intell. Veh.*, vol. 5, no. 1, pp. 7–20, 3 2020.

[24] F. Molinari, N. N. Anh, and L. Del Re, "Efficient mixed integer programming for autonomous overtaking," *Proc. Am. Control Conf.*, pp. 2303–2308, 2017.

[25] R. Quirynen, S. Safaoui, and S. Di Cairano, "Real-time Mixed-Integer Quadratic Programming for Vehicle Decision Making and Motion Planning," *arXiv preprint arXiv:2308.10069*, 8 2023.

[26] R. Quirynen and S. Di Cairano, "Tailored presolve techniques in branch-and-bound method for fast mixed-integer optimal control applications," *Optimal Control Appl. Methods*, vol. 44, no. 6, pp. 3139–3167, 11 2023.

[27] S. S. Lodhi, N. Kumar, and P. K. Pandey, "Autonomous vehicular overtaking maneuver: A survey and taxonomy," *Veh. Commun.*, vol. 42, p. 100623, 2023.

[28] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An Open Urban Driving Simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, 11 2017, pp. 1–16.

[29] P. Polack, F. Altche, B. D'Andrea-Novel, and A. de La Fortelle, "The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?" in *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 6 2017, pp. 812–818.

[30] M. Werling, J. Ziegler, S. Kammel, and S. Thrun, "Optimal trajectory generation for dynamic street scenarios in a frenét frame," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2010, pp. 987–993.

[31] D. Kloeser, T. Schoels, T. Sartor, A. Zanelli, G. Prison, and M. Diehl, "NMPC for Racing Using a Singularity-Free Path-Parametric Model with Obstacle Avoidance," in *IFAC-PapersOnLine*, vol. 53, no. 2. Elsevier B.V., 2020, pp. 14 324–14 329.

[32] L. A. Wolsey, *Integer Programming*, 2nd ed. John Wiley & Sons, 2020.

[33] B. Alrifaee, "Networked Model Predictive Control for Vehicle Collision Avoidance," Ph.D. dissertation, RWTH Aachen University, 2017.

[34] F. Janeček, M. Klaučo, M. Kalúz, and M. Kvasnica, "OPTIPLAN: A Matlab Toolbox for Model Predictive Control with Obstacle Avoidance," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 531–536, 2017.

[35] A. Richards and J. How, "Mixed-integer programming for control," *Proc. Am. Control Conf.*, vol. 4, pp. 2676–2683, 2005.

[36] D. Ioan, I. Prodan, S. Olaru, F. Stoican, and S. I. Niculescu, "Mixed-integer programming in motion planning," *Annu. Rev. Control*, vol. 51, no. October 2020, pp. 65–87, 2021.

[37] Y. Puranik and N. V. Sahinidis, "Domain reduction techniques for global NLP and MINLP optimization," *Constraints*, vol. 22, no. 3, pp. 338–376, 2017.

[38] M. Werling, S. Kammel, J. Ziegler, and L. Gröll, "Optimal trajectories for time-critical street scenarios using discretized terminal manifolds," *The Int. J. Robotics Research*, vol. 31, no. 3, pp. 346–359, 3 2012.

[39] J. V. Frasch, A. Gray, M. Zanon, H. J. Ferreau, S. Sager, F. Borrelli, and M. Diehl, "An auto-generated nonlinear MPC algorithm for real-time obstacle avoidance of ground vehicles," in *2013 European Control Conference, ECC 2013*. IEEE Computer Society, 2013, pp. 4136–4141.

[40] A. Richards, T. Schouwenaars, J. P. How, and E. Feron, "Spacecraft trajectory planning with avoidance constraints using mixed-integer linear programming," *J. Guid. Control. Dyn.*, vol. 25, no. 4, pp. 755–764, 2002.

[41] J. Ploeg, N. Van De Wouw, and H. Nijmeijer, "Lp string stability of cascaded systems: Application to vehicle platooning," *IEEE Trans. on Control Syst. Technol.*, vol. 22, no. 2, pp. 786–793, 2014.

[42] G. J. Naus, R. P. Vugts, J. Ploeg, M. J. Van De Molengraft, and M. Steinbuch, "String-stable CACC design and experimental validation: A frequency-domain approach," *IEEE Trans. on Veh. Technol.*, vol. 59, no. 9, pp. 4268–4279, 2010.

[43] C. Liu, W. Zhuang, G. Yin, Z. Huang, and H. Liu, "A Survey on Cooperative Longitudinal Motion Control of Multiple Connected and Automated Vehicles," *IEEE Intell. Transp. Syst. Mag.*, vol. 12, no. 1, pp. 4–24, 2020.

[44] S. Feng, Y. Zhang, S. E. Li, Z. Cao, H. X. Liu, and L. Li, "String stability for vehicular platoon control: Definitions and analysis methods," *Annu. Rev. Control*, vol. 47, pp. 81–97, 2019.

[45] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[46] J. Schlechtriemen, K. P. Wabersich, and K. D. Kuhnert, "Wiggling through complex traffic: Planning trajectories constrained by predictions," *IEEE Intell. Veh. Symp. Proc.*, vol. 2016-Augus, no. Iv, pp. 1293–1300, 2016.

[47] A. L. Gratzer, A. Schirrer, E. Thonhofer, F. Pasic, S. Jakubek, and C. Mecklenbrauker, "Short-Term Collision Estimation by Stochastic Predictions in Multi-Agent Intersection Traffic," in *2022 International Conference on Electrical, Computer and Energy Technologies (ICECET)*, no. June. IEEE, 7 2022, pp. 1–6.

[48] Z. Ding and H. Zhao, "Incorporating Driving Knowledge in Deep Learning Based Vehicle Trajectory Prediction: A Survey," *IEEE Trans. on Intell. Veh.*, vol. 8, no. 8, pp. 3996–4015, 2023.

[49] A. L. Gratzer, A. Schirrer, and S. Jakubek, "Agile Multi-Agent Model Architecture for Intelligent Intersection Traffic Simulation," *IFAC-PapersOnLine*, vol. 55, no. 27, pp. 89–95, 2022.

[50] Gurobi Optimization LLC, "Gurobi Optimizer Reference Manual," 2021.

[51] M. Treiber and A. Kesting, *Traffic Flow Dynamics - Data, Models and Simulation*. Springer, 2013.

[52] G. M. Hoffmann, C. J. Tomlin, M. Montemerlo, and S. Thrun, "Autonomous automobile trajectory tracking for off-road driving: Controller design, experimental validation and racing," *Proc. Am. Control Conf.*, pp. 2296–2301, 2007.

[53] R. Gutierrez, J. F. Arango, C. Gomez-Huelamo, L. M. Bergasa, R. Barea, and J. Araluce, "Validation method of a self-driving architecture for unexpected pedestrian scenario in CARLA simulator," *IEEE Intell. Veh. Symp. Proc.*, vol. 2021-July, no. Iv, pp. 1144–1149, 2021.

[54] D. R. Morais and A. P. Aguiar, "Model Predictive Control for Self Driving Cars: A Case Study Using the Simulator CARLA within a ROS Framework," in *2022 IEEE International Conference on Autonomous Robot Systems and Competitions, ICARSC 2022*. Institute of Electrical and Electronics Engineers Inc., 2022, pp. 124–129.

[55] Z. Zhou, C. Rother, and J. Chen, "Event-Triggered Model Predictive Control for Autonomous Vehicle Path Tracking: Validation Using CARLA Simulator," *IEEE Trans. on Intell. Veh.*, vol. 8, no. 6, pp. 3547–3555, 6 2023.

[56] P. Kaur, S. Taghavi, Z. Tian, and W. Shi, "A Survey on Simulators for Testing Self-Driving Cars," *Proc. - 2021 4th Int. Conf. on Connect. Auton. Driving, MetroCAD 2021*, pp. 62–70, 2021.

Alexander L. Gratzer (Student Member, IEEE) received the M.Sc. degree in mechanical engineering from TU Wien, Vienna, Austria, in 2019 and currently works toward the Ph.D. degree. Since 2019, he has been a member of the project team with the Institute of Mechanics and Mechatronics, TU Wien. His research interests include modeling, control, and optimization of complex systems with a recent focus on automated driving and intelligent transportation systems (ITS).

Alexander Schmiedhofer received the M.Sc. degree in mechanical engineering from TU Wien, Vienna, Austria, in 2024 and currently works toward the Ph.D. degree.

Alexander Schirrer received the M.S. degree in mechanical engineering, the Ph.D. degree, and the Habilitation from TU Wien, Vienna, Austria, in 2007, 2011, and 2018, respectively. Since 2011, he has been a Postdoctoral Researcher and Teacher of graduate-level lectures with the Institute of Mechanics and Mechatronics, TU Wien. His research interests include modeling, simulation, optimization, and control of complex and distributed-parameter systems.

Stefan Jakubek received the M.S. degree in mechanical engineering, the Ph.D. degree, and the Habilitation from TU Wien, Vienna, Austria in 1997, 2000, and 2007, respectively. From 2007 to 2009, he was the Head of Development for Hybrid Powertrain Calibration and Battery Testing Technology with AVL List GmbH, Graz, Austria. He is currently a Professor at the Institute of Mechanics and Mechatronics, TU Wien. His research interests include fault diagnosis and system identification.