Model Predictive Control for Lane Merging Automation with Recursive Feasibility Guarantees [⋆]

M.E. Geurts [*] A. Katriniok [*,**] E. Silvas [*,***] W.P.M.H. Heemels [*]

[*] *Department of Mechanical Engineering, Control Systems Technology Section, Eindhoven University of Technology, The Netherlands*
{m.e.geurts,a.katriniok,e.silvas,m.heemels}@tue.nl.
[**] *Ford Research & Innovation Center (RIC), Aachen, Germany*
[***] *Department of Integrated Vehicle Safety, TNO, Helmond, The Netherlands*

Abstract: In order to make the complex driving task of merging safer, in this paper we consider the automated merging of an autonomous vehicle into a mixed-traffic flow scenario (i.e., traffic including autonomous and manually driven vehicles). In particular, we propose a novel MPC-based algorithm to perform a merging procedure from a double lane into a single lane and continue with (adaptive) cruise control ((A)CC) functionality after the merge. The proposed MPC balances fast progress along the path with comfort, while obeying safety and maximum allowed velocity bounds. Recursive feasibility, leading to safety and proper behavior, is guaranteed by the design of a proper terminal set, extending existing ones in the literature. The on-line MPC problem is translated into a mixed integer quadratic program (MIQP) that can be solved for global optimality. Through numerical simulations we demonstrate the behavior and effectiveness of the proposed MPC merging scheme.

Keywords: Autonomous vehicles; Predictive control; Trajectory Tracking and Path Following; Nonlinear and optimal automotive control; Safety.

1. INTRODUCTION

Automated vehicles come with the promise that they will improve mobility and transportation systems, making them safer, more comfortable, more efficient and more sustainable. Additionally, for an increasing level of automation (Level 4+) (SAE, 2018) they will release the drivers from their driving tasks and allow them to perform secondary activities during the drive. Bringing these vehicles on the road will require a transition where autonomous vehicles will coexist with manually driven vehicles. As a result, one of the greatest challenges identified in the literature is to plan safe and optimal trajectories for automated vehicles in complex and uncertain situations, with prior unknown movements of manually driven vehicles (Khonji et al., 2020; Wang et al., 2020). To predict future movements of manually driven vehicles, combined data-driven and model based methods could be used. However, given these predictions, the challenge remains to find an optimal trajectory and a control strategy for the automated Level 4+ vehicle in every road situation (van Nunen et al., 2017; Paden et al., 2016).

Ideally, for Level 4+ systems, where drivers are not a backup anymore, trajectory generation and control algorithms should be scalable (i.e., function well in a broad mix of scenarios), generalizable (i.e., apply also in new and unknown scenarios), capable of dealing with manually driven vehicles, and benefit (if possible and available) from other vehicles' connectivity and cooperation capabilities. In the development of Level 4+ systems, complex scenarios (e.g., scenarios with high traffic density and a lot of uncertainty) can pose major challenges in mixed-traffic (i.e., traffic including autonomous and manually driven vehicles) and can cause dangerous situations (Bouton et al., 2019, 2020). One of these complex scenarios is lane merging, which is the focus of this paper.

Merging into a dense traffic flow is a complex driving task, also for human drivers, as they have to perform multiple tasks at once (e.g., cognition (detecting empty space), decision making (decide where to merge) and driving operations (drive car to gap)) (Nagahama et al., 2021; Cao et al., 2019). This is also true for an autonomous vehicle, due to uncertain measurements of the surrounding vehicles, unknown intentions, unknown future trajectories of other vehicles and interaction with other road users (Hubmann et al., 2018; Bouton et al., 2020). When the merging algorithm is not proactively able to make a decision in time and find a gap, the autonomous vehicle can freeze and stop functioning (Bouton et al., 2019, 2020), with all corresponding risks involved.

In order to make merging safer, we present a new model predictive control (MPC) algorithm for merging. While developing the MPC algorithm, the focus is on safety guarantees through a rigorous proof of recursive feasibility and allowing good overall driving behavior. This resulting algorithm is able to perform the merging procedure from a double lane into a single lane. It continues in an integrated manner, with adaptive cruise control (ACC) functionality after the merge, all while guaranteeing an appropriate time headway, which changes based on the stage in the merging process (i.e., the ego vehicle in initial merging lane, crossing the lane markings or being in the final lane). To guarantee recursive feasibility of the MPC scheme at any time we propose an appropriate control invariant terminal set, which

[⋆] This work is part of the research program AMADeuS with project number 18489, which is partly financed by the Netherlands Organisation for Scientific Research (NWO).

consists of the union of two disjoint sets based on merging before or after the vehicle in the target lane. This forms an enlargement of the terminal set in literature and avoids issues that would be experienced with the set as proposed in Bali et al. (2018), see Sec. 2 for a detailed discussion of related work. The MPC is translated into a mixed integer quadratic program (MIQP) that can be solved for global optimality. Through numerical simulations we demonstrate the behavior and effectiveness of the proposed merging scheme.

The remainder of this paper is as follows. Related work is presented in Sec. 2. The problem and model setup for the merging scenario is introduced in Sec. 3. In Sec. 4 we propose our MPC for lane merging. In Sec. 5 we show the strength of our new method in a numerical case study. This paper ends with the conclusions and future work in Sec. 6.

2. RELATED WORK

The complex task of lane merging has received quite some attention in recent years and different solutions and approaches were proposed. In Bouton et al. (2020, 2019); Wang et al. (2021) the problem of merging into dense traffic with autonomous vehicles is solved by using reinforcement learning (RL) and game theory. A drawback of RL might be that the behavior of the controller is unpredictable in untrained situations, and given the safety-critical nature of automated driving more transparent explainable methods with safety guarantees are desirable. In Hubmann et al. (2018) local path planning is considered, with merging an autonomous vehicle in an urban environment with high traffic density. The problem is formulated as a Partially Observable Markov Decision Process (POMDP), which includes the uncertain behavior of surrounding drivers in the state space with a motion model learned from real-world data. The result is an algorithm that generates the path the vehicle should take. However, no trajectory generation and control is considered explicitly in Hubmann et al. (2018), while this forms the focus in this paper.

Finally, several MPC-based solutions are available for autonomous driving. For example, in Naus et al. (2010) an ACC solution is proposed. However, the MPC-based ACC is not shown to be recursively feasible, and in case infeasibility occures, the driving authority is giving back control to the driver. This contrasts our research, where we prove recursive feasibility for the full merging scenario including the ACC functionality after the merge. Further, in Katriniok et al. (2017); Schweidel et al. (2022) MPC is used to handle intersection automation, Ioan et al. (2021) uses it for motion planning using mixed integer programming (MIP), Dixit et al. (2018) for overtaking with potential fields, Alcalá et al. (2019) for trajectory tracking of linear parameter varying (LPV) dynamic systems and Quirynen et al. (2020) for obstacle avoidance. MPC is used to develop trajectory generation in merging scenarios as well in Cao et al. (2019); Bali et al. (2018); Mukai et al. (2017). In Cao et al. (2019) the focus is on robustness enhancement and handling sensor noise. In the corresponding MPC-development process, a cooperative maneuver between two vehicles was developed, different from our solution which focuses only on the ego vehicle behavior, while including knowledge of the other road users. In Mukai et al. (2017) an MPC-MIP merging path generation is developed in a two-dimensional coordinate system, while guaranteeing safety with predicting the other road users' behavior based on the initial state, car length and resulting available gaps, without an explicit proof of recursive feasibility. This is in contrast to our one-dimensional merging automation, which includes full knowledge of target vehicles, maintaining a safe time headway at anytime and a proof of recursive feasibility. In Bali et al. (2018) a *multi-vehicle* MPC-MIQP controller for merging at junctions is developed, different from our ego vehicle lane merging automation. Differences are that the MPC controller in Bali et al. (2018) solves a centralized multi-vehicle solution, which incorporates all road users while giving priority to certain vehicles, while in our research only the solution for the ego-vehicle is determined, with full knowledge of the other road users behavior and relative states. Further, collision prevention is ensured in Bali et al. (2018) by using collision sets, in contrast to our merging position-dependent safe following distance. Similar as in this paper, Bali et al. (2018) shows the recursive feasibility of the MPC-solution using time-headway conditions on the vehicle distance. However, Bali et al. (2018) shows recursive feasibility for a static obstacle and argues that this holds for forward driving vehicles too. Different from our proof, which includes directly dynamic obstacles, which leads to a more explicit and practically less restrictive proof in case the other road users' velocity is non-negative.

In conclusion, to the authors' best knowledge, merely no methods with proved feasibility exists for finding a trajectory in lane merging scenarios with dynamic objects, which can also scale to other scenarios, i.e., generalize to more situations (merging and ACC). The majority of previous methods rely on transitioning the vehicle control to the human driver when no feasible path is available or only prove the recursive feasibility for the merging process in a specific situation. This can lead to dangerous situations, in the case where no feasible path is found. Finding and executing a feasible path should be done completely by the system in higher levels of vehicle automation. To handle this complex task of merging in a fully automated way, a novel method is needed.

3. PROBLEM FORMULATION & MODEL

In this section we discuss the merging use case, the vehicle architecture, the adopted vehicle model and constraints, and the problem formulation considered in this paper.

3.1 Use case description

In this research the objective is to develop a controller for an autonomous vehicle, that should be able to execute the following scenario (see Fig. 1):

(I) *Initial position*: The ego vehicle (Agent 1, grey vehicle in Fig. 1) is driving on the ego lane. The target vehicle (Agent 2, white car in Fig. 1) is driving on the target lane.
(II) *Goal*: The ego lane is closing off, requiring the ego vehicle to merge into the target lane (with lane change point LC, the fixed position where ego vehicle is at the target lane for the first time), before a certain merging point MP.
(III) *Final position*: The ego vehicle is driving on the target lane, behind or in front of the target vehicle with a safe relative distance (i.e., essentially using adaptive cruise control (ACC) or cruise control (CC) functionality, respectively).

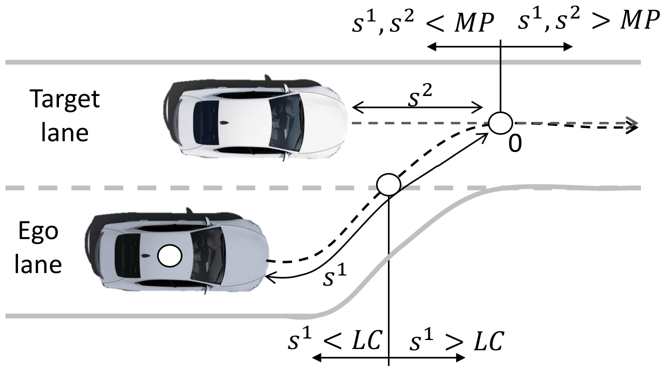

Fig. 1. Set-up of the lane-merging scenario.

3.2 Architecture of the autonomous vehicle

In Fig. 2 a high-level overview is shown of the control system architecture needed for carrying out the merging manoeuvre. In this figure, light grey blocks indicate algorithms assumed to be already available, being a world model, a trajectory prediction of other vehicles, a scenario generator, a path generator and low-level vehicle controls. The white block, the block that is developed in this paper, is a trajectory generation algorithm and a vehicle control algorithm. In fact, we propose a model predictive control (MPC) approach to provide an integrated solution for trajectory generation and longitudinal vehicle control, under the following assumptions:

(I) the path of the ego vehicle is fixed by a path generator and exactly followed (i.e., the positions the vehicle will travel during the merging process are given, but the speed of traveling for the ego vehicle along the path is not known and has to be determined);
(II) the future trajectory (i.e., position and velocity at any time step) of the target vehicle (i.e., the other road users) is known to the ego vehicle over the prediction horizon;
(III) the target vehicle is driving with a constant velocity, as long as the ego vehicle is not preventing the target vehicle from moving freely (i.e., when ego vehicle moves in front of target vehicle the target vehicle starts to behave in an ACC mode).

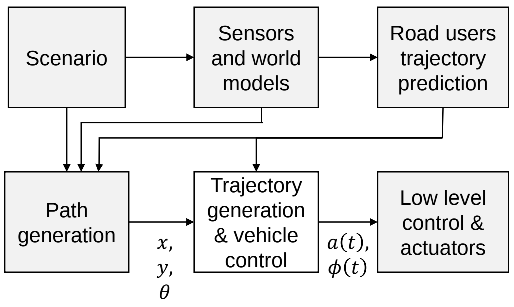
Scenario
Sensors and world models
Road users trajectory prediction
Path generation
x, y, θ
Trajectory generation & vehicle control
$a(t)$, $\phi(t)$
Low level control & actuators

Fig. 2. Overview of functions needed in the lane-merging scenario, with light grey blocks assumed available and the white block that has to be determined by our envisioned MPC scheme.

3.3 Vehicle models

In this merging scenario, the dynamics of the ego and the target vehicle (i.e., Agent 1 and 2, respectively) are formulated in terms of the one-dimensional path coordinate s^i_k with respect to the center of vehicle $i \in \{1,2\}$ to the origin of the coordinate system, the merging point MP, i.e., $s^1_k = 0$ indicates that Agent 1 is at the merging point at time step $k \in \mathbb{N}_0$; velocity v^i_k (for Agent $i \in \{1,2\}$); relative velocity $\Delta v_k = v^2_k - v^1_k$; relative distance $\Delta s_k = s^2_k - s^1_k$ and input acceleration u_k of the ego vehicle, all at discrete time $k \in \mathbb{N}_0$ (see Fig. 1). Discrete time $k \in \mathbb{N}_0$ corresponds to real time $t = kT_s$, where T_s denotes the sampling period. By selecting the state as $x_k = \left[\Delta s_k \ \Delta v_k \ s^1_k \ v^1_k\right]^T \in \mathbb{R}^{n_x}$ with $n_x = 4$, we can obtain the relevant dynamics by exact discretization of the double integrator dynamics, leading to

$$x_{k+1} = Ax_k + Bu_k \tag{1}$$

with

$$A = \begin{bmatrix} 1 & T_s & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T_s \\ 0 & 0 & 0 & 1 \end{bmatrix} \text{ and } B = \begin{bmatrix} -\frac{1}{2}T_s^2 \\ -T_s \\ \frac{1}{2}T_s^2 \\ T_s \end{bmatrix},$$

where we made the modelling assumption in the dynamics that the target vehicle (Agent 2) drives at constant speed. Future work will include extensions towards the inclusion of varying speeds

Note that, the dynamics are formulated in a one-dimensional frame, with the distance towards MP. This is possible due to the assumption that the ego vehicle path is exactly followed.

3.4 Vehicle constraints

For the ego vehicle, there are actuator limitations in terms of minimal and maximal acceleration, given by $u_{min} < 0 < u_{max}$. Hence, we must ensure that

$$u_{min} \leqslant u_k \leqslant u_{max} \text{ for all } k \in \mathbb{N}_0. \tag{2}$$

Moreover, there are constraints for the maximally allowed speed of the ego vehicle, v_{max}, which can vary from one road segment to another, but for the considered merging scenario, we assume it to be constant. Further, it is assumed that the speed is always non-negative (i.e., vehicles are only driving forward or stand still). Hence,

$$0 \leqslant v^1_k \leqslant v_{max} \text{ for all } k \in \mathbb{N}_0. \tag{3}$$

3.5 Control requirements and problem formulation

In the lane merging scenario it is desired that the following requirements are met:

(i) Agent 1's speed v^1 should be as close as possible to the desired speed v^1_{ref} with v^1_{ref} being set as, e.g., in a cruise controller;
(ii) the merging behavior should be performed as efficiently as possible in terms of fuel consumption (considered here as to minimizing accelerations);
(iii) and achieving comfortable driving behavior (by minimizing step changes of control input related to jerk).

The objective of the paper can now be formulated as the design of a control algorithm that solves the trajectory generation and vehicle control problems in Fig. 2 in an integrated fashion for the merging scenario described in Sec. 3.1 (including the safety constraints, see Sec. 4.2) that meets the requirements (i)-(iii) above, and adheres to the constraints (2) and (3) in Sec. 3.4.

4. CONTROLLER DESIGN

To develop an integrated solution for the trajectory generation and the vehicle control, an MPC-based approach is used. As a prediction model, we adopt the dynamics in (1). The cost function and constraint sets for the MPC are defined below.

4.1 *Cost function*

To incorporate the desired requirements as defined in Sec. 3.5, we propose the quadratic objective function for Agent 1 as

$$J(x_k, U_k) := Q \sum_{j=1}^{N} (v^1_{ref} - v^1_{k+j|k})^2 \tag{4}$$
$$+R \sum_{j=0}^{N-1} (\Delta u_{k+j|k})^2 + S \sum_{j=0}^{N-1} (u_{k+j|k})^2$$

with $x_{k|k} = x_k$ being the current state at time step $k \in \mathbb{N}_0$, $U_k = [u_{k|k}, \dots, u_{k+N-1|k}]^T$ denoting the predicted control sequence of the agent over the control horizon with $u_{k+j|k}$ the prediction of u_{k+j} made at time step k and $\Delta u_{k+j|k} = u_{k+j|k} - u_{k+j-1|k}$ the predicted increment of the control input. The constants $Q > 0$, $R > 0$ and $S > 0$ are positive weighting coefficients, for the desired requirement (i) (i.e., speed as close as possible to the desired speed), (iii) (i.e., achieving comfortable behavior by minimizing jerk) and (ii) (i.e., efficient fuel consumption by minimizing accelerations) in Sec. 3.5, respectively. Clearly, if one of the weighting coefficients is larger than the other, the cost function will give more priority to that desired requirement. Finally, $N \in \mathbb{N}$ denotes the prediction horizon.

4.2 *Constraints*

The limitations on the deceleration and acceleration (as discussed in Sec. 3.4) result in the input constraint:

$$u_{k+j|k} \in \mathbb{U} := \{u \in \mathbb{R} | u_{min} \leqslant u \leqslant u_{max}\} \tag{5}$$

for $j \in \{0, 1, \dots, N-1\}$. The limitations on the vehicle speed lead to the following state constraint along the prediction horizon:

$$x_{k+j|k} \in \mathbb{X} := \{x \in \mathbb{R}^4 | 0 \leqslant v^1 \leqslant v_{max}\} \tag{6}$$

where v_{max} dictates the maximal allowable speed and $j \in \{1, 2, \dots, N\}$.

Besides the hard state constraint (6), to guarantee safety, we will include another hard state constraint, to enforce that Agent 1 keeps a safe (varying) distance d_{safe} from Agent 2 to avoid collisions, when Agent 1 is behind Agent 2. When Agent 1 is in front of Agent 2, we assume that Agent 2 will take care of this safety distance (see assumption (3) in Sec. 3.2). Note that an aggressive cut-in will not occur, since $|\Delta s| \geqslant 0$ before Agent 1 is in front of Agent 2.

In fact, we will formulate the safety condition as a constraint of the form $|\Delta s| \geqslant d_{safe}$ on the relative distance between the vehicles. For proper merging and safety behavior d_{safe} will depend on the situation and thus on the states during the merge. In particular, we let d_{safe} depend on the position s^1 of Agent 1, relative to the merging point MP and the lane change point LC, i.e., the point where Agent 1 crosses the line to the target lane (see Fig. 1) and also on its velocity v^1 (to introduce an appropriate time headway (Swov,a, 2022; Swov,b, 2022)). Hence, $d_{safe} = d_{safe}(s^1, v^1)$ is a function of the state variables s^1 and v^1, and we propose to define it as follows:

(I) In case Agent 1 has not yet started the lane change manoeuvre (i.e., when $s^1 < LC$), then Agent 1 and Agent 2 are still driving in separate lanes, with a safe lateral distance, due to their paths being fixed. In this situation we define $d_{safe}(s^1, v^1) := 0[m]$.

(II) In case Agent 1 started the lane change procedure, but is not completely merged (i.e., $LC < s^1 < MP$), then Agent 1 and 2 will partly drive in the same lane, therefore a safe longitudinal distance towards Agent 2 is needed. To accomplish this, we define $d_{safe}(s^1, v^1) := v^1[m]$, to guarantee a speed-dependent safety gap (i.e., a time headway of $1[s]$)(Roelofsen, 2009).

(III) In case Agent 1 passed the merging point (i.e., $s^1 > MP$), then Agent 1 and 2 drive in the same lane. This calls for a safe longitudinal distance in case Agent 1 is behind Agent 2 (e.g., as seen in ACC functionality)(Naus et al., 2010). Therefore, in this situation we set $d_{safe}(s^1, v^1) := 2v^1[m]$ to have a time headway of $2[s]$, which is often used in ACC settings (Swov,a, 2022).

These constraints can be combined into the safety time headway constraint:

$$|\Delta s| \geqslant d_{safe}(s^1, v^1) = \begin{cases} 2v^1 & \text{if} \quad s^1 > MP \quad \wedge\ s^2 > s^1 \\ v^1 & \text{if}\ LC < s^1 \leqslant MP \wedge s^2 > s^1 \\ 0 & \quad \text{otherwise.} \end{cases} \tag{7}$$

4.3 *Terminal set*

In order to guarantee recursive feasibility, meaning that if the MPC optimization problem is feasible at time $k \in \mathbb{N}_0$, then it is also feasible at the next time step $k+1$, we will design an appropriate terminal set. In particular, if the terminal set is chosen to be controlled invariant for the dynamics (1), recursive feasibility is guaranteed (Rawlings, 2009).

Definition 1. (Rawlings, 2009) A set $\mathbb{X}_T$ is said to be controlled invariant for the system $x_{k+1} = Ax_k + Bu_k$ with input constraint set $\mathbb{U}$, if for all $x_k \in \mathbb{X}_T$ there is an input $u_k \in \mathbb{U}$ such that $Ax_k + Bu_k \in \mathbb{X}_T$.

The controlled invariant terminal set $\mathbb{X}_T$ for our merging MPC is the union of two disjoint sets that depend on whether Agent 1 merged in front or behind Agent 2, i.e., $\mathbb{X}_T = \Omega_1 \cup \Omega_2$ with Ω_1 being a subset of the set described when $\Delta s \geqslant 0 \wedge s^1 \geqslant MP$ (i.e., a merge behind) and Ω_2 a subset of $\Delta s \leqslant 0 \wedge s^1 \geqslant MP$ (i.e., a merge in front). In the vehicle dynamics it is assumed that Agent 2 has a constant velocity and is not standing still before the merging point; therefore, this invariant terminal set is valid for a lane merging scenario of 2 vehicles, where the target vehicle is a dynamic obstacle with constant speed. Note that the terminal set conditions are defined in such a way that this obstacle has zero acceleration. In future work, we are interested in defining this set for a situation where the target vehicle is not driving at a constant speed. Due to space limitations the proof is omitted.

Theorem 2. The set

$$\Omega_1 = \{\begin{pmatrix}\Delta s & \Delta v & s^1 & v^1\end{pmatrix}^T \in \mathbb{R}^4 | \Delta s \geqslant 2v^1,$$
$$\Delta v \geqslant 2u_{min}, 0 \leqslant v^1 \leqslant v_{max}, s^1 \geqslant MP\}$$

is controlled invariant for system (1) with input constraint set $[u_{min}, u_{max}]$, when $u_{min} < 0 \leqslant u_{max}$, time step $0 < T_s \leqslant 2$,

$2 \geqslant \frac{-v_{max}+v_k^2}{u_{min}}$ and target car constant velocity at time k is $v_k^2 \geqslant 0$.

We select the set Ω_2 as

$$\Omega_2 = \{\left(\Delta s\ \Delta v\ s^1\ v^1\right)^T \in \mathbb{R}^4 | \Delta s \leqslant 0, \\ 0 \leqslant v^1 \leqslant v_{max}, s^1 \geqslant MP\}.$$

This set is used in case Agent 1 merges in front of Agent 2 ($\Delta s \leqslant 0$), at the end of the prediction horizon. In usual traffic behaviour, the safety constraint has to be maintained by Agent 2 in this scenario, however Agent 1 ensures a non-aggressive cut-in due to safety constraints when it is still behind Agent 2. Therefore, we will use Ω_2 as part of the terminal set $\mathbb{X}_T = \Omega_1 \cup \Omega_2$ under the assumption that Agent 2 ensures keeping a sufficiently large safety distance, e.g., by adjusting its behavior according to an ACC functionality, thereby leading to recursive feasibility, if we use $\mathbb{X}_T$ as a terminal set. Note that at the moment of the actual merge of Agent 1 in front of Agent 2, (7) is in place enforcing a larger gap than just $\Delta s \leqslant 0$. In addition, the safety constraint (7) can be abandoned after the actual merge has taken place and Agent 1 is in front of Agent 2.

In conclusion, in order to guarantee recursive feasibility in our MPC scheme the following constraint is defined at the end of the prediction horizon:

$$x_{N|k} \in \mathbb{X}_T = \Omega_1 \cup \Omega_2. \tag{8}$$

Note that to prevent the solution being unintentionally infeasible before the merging point, the prediction horizon and time step should be of sufficient length to incorporate the non-feasibility before MP. In practice this will mean that the functionality is activated when the car is sufficiently close to MP.

Note that this terminal set $\mathbb{X}_T$ is in general not the maximal controlled invariant set, inside $\mathbb{X}$ and given the allowable control actions (Kerrigan et al., 2002). However, note that the terminal set in (8) is larger than the set proposed by Bali et al. (2018), in case $v_k^2 > -\frac{1}{2}T_s u_{min}$. The (Bali et al., 2018)-set, is given in the next theorem:

Theorem 3. (Bali et al., 2018) The set

$$\Omega_3 = \{\left(\Delta s\ \Delta v\ s^1\ v^1\right)^T \in \mathbb{R}^4 | \Delta s \geqslant t_h v^1, 0 \leqslant v^1 \leqslant v_{max}\}$$

with $t_h > 0$ the time headway, is controlled invariant for system (1) with input constraint set $[u_{min}, 0]$, when $u_{min} < 0$, if time step $0 < T_s \leqslant 2t_h$, $t_h \geqslant \frac{v_{max}}{-u_{min}} - \frac{T_s}{2}$ and target car constant velocity at time k is $v_k^2 = 0[\frac{m}{s}]$.

When terminal sets (8) and Ω_3 in Theorem 3 are compared with each other, it can be seen that (8) restricts the maximum relative velocity Δv while Ω_3 restricts the maximum velocity v_1. This means that our propsed terminal set has a larger maximum allowed velocity set and terminal set, in case $v_k^2 > -\frac{1}{2}T_s u_{min}$, compared to Ω_3. Hence, applying Ω_3 restricts the practical applicability of the control scheme to low speeds while leveraging our proposed terminal set (8) does not.

4.4 *Complete MPC algorithm*

Assembling all the ingredients above, leads to the following MPC problem for state x_k at time $k \in \mathbb{N}_0$:

$$\begin{aligned} &\min_{U_k} J(x_k, U_k) \\ &\text{s.t. } x_{k+j+1|k} = Ax_{k+j|k} + Bu_{k+j|k},\ j = \{0, 1, \ldots, N-1\}, \\ &\quad x_{k|k} = x_k, \\ &\quad x_{k+j|k} \in \mathbb{X}, \quad j = \{0, 1, \ldots, N-1\}, \\ &\quad u_{k+j|k} \in \mathbb{U}, \quad j = \{0, 1, \ldots, N-1\}, \\ &\quad x_{k+N|k} \in \mathbb{X}_T, \\ &\quad |\Delta s_{k+j|k}| \geqslant d_{safe}(s^1_{k+j|k}, v^1_{k+j|k}), j = \{0, 1, \ldots, N-1\}. \end{aligned} \tag{9}$$

Let $U_k^* = [u^*_{k|k}, u^*_{k+1|k}, \ldots, u^*_{k+N-1|k}]$ be a minimizer of (9) at time k. The control input applied to the system (1) at time $k \in \mathbb{N}_0$ is then $u_k = u^*_{k|k}$ for $k \in \mathbb{N}_0$. Note that existence of a minimizer is in general guaranteed due to the fact that the optimization problem can be reformulated as a MIQP with positive definite cost following ideas from Bemporad et al. (1999).

Note that by the construction of our terminal set, and under the stated assumptions, the MPC scheme is recursively feasible, i.e., if (9) is feasible for x_k at time step $k \in \mathbb{N}_0$, then the MPC problem is feasible at time step $k+1$ for the new state $x_{k+1} = Ax_k + Bu^*_{k|k}$.

5. NUMERICAL EXAMPLE

The behavior of the MPC-MIQP controller is verified and compared to Bali et al. (2018) with a numerical example, which could for example be witnessed during a traffic jam or urban traffic scenario. In this example we will verify if desirable behavior is obtained in two scenarios when Agent 1 merges behind or in front of Agent 2, respectively. The following parameters are selected: $v_{ref} = \frac{50}{3.6}[\frac{m}{s}]$ (according to an urban scenario), $v_{max} = 1.1 \cdot v_{ref}$, $T_s = 0.2[s]$, $N = 50$ (related to a horizon of $10[s]$ to cover the whole lane merging scenario), $Q = 1$, $R = 1$, $S = 1$ (to balance all desired requirements), $u_{min} = -3[\frac{m}{s^2}]$, $u_{max} = 5[\frac{m}{s^2}]$, $MP = 0[m]$, $LC = -15[m]$, initial positions $s_0^1 = -150[m]$, $s_0^2 = -144[m]$ (i.e., the distance towards MP) and $\Delta s_0 = s_0^2 - s_0^1$. Further, in the terminal set $v_{max,T} = v_0^2 - 2u_{min}[m/s]$ and $v_{max,T} = -u_{min}(\frac{1}{2}T_s + 2)[m/s]$ for the set defined in (8) and Theorem 3, respectively. [1]

5.1 *Scenario 1: Agent 1 merges behind Agent 2*

In order to verify the behavior of Agent 1, when it merges behind Agent 2, the initial velocities $v_0^1 = 12.5[\frac{m}{s}]$, $v_0^2 = 12[\frac{m}{s}]$ and $\Delta v_0 = v_0^2 - v_0^1[\frac{m}{s}]$ are selected. The MPC-MIQP controllers results are visualised in Fig. 3. These results are obtained in a maximum computation time of $0.4708[s]$ each time step [1], indicating that the sampling time of $T_s = 0.2[s]$ is currently too small for real world implementation. It is part of future work to address this computational time, although we expect that with a dedicated computation hardware our algorithm can meet the real-time specification.

Agent 1 starts on the ego lane behind (in longitudinal direction) Agent 2 in the merging lane. However, Agent 1's velocity is slightly larger than the velocity of Agent 2. It is not possible to accelerate sufficiently to merge in front of Agent 2. Therefore, the optimal solution is that Agent 1 decelerates and then

[1] The numerical simulation has been executed with Gurobi MIQP solver in Matlab R2022b on a Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz.

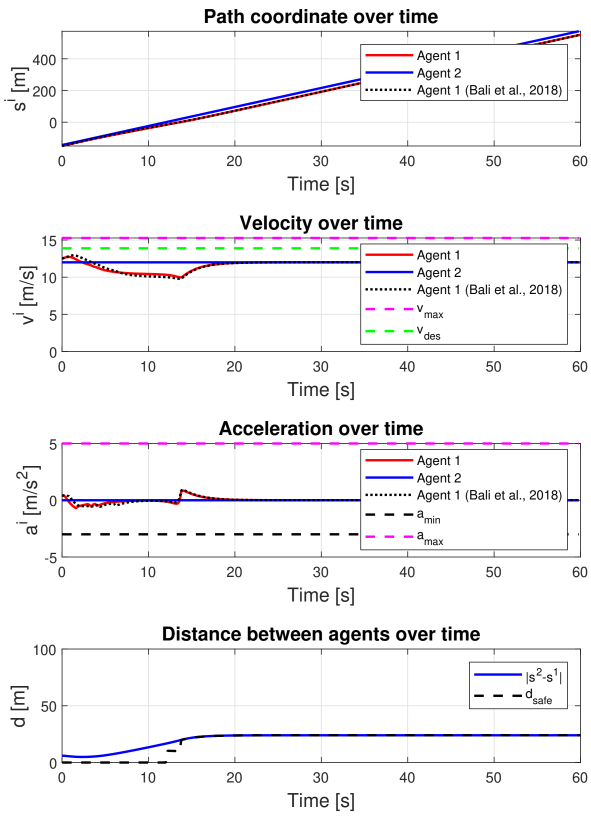

Fig. 3. Merging behind Agent 2 behavior, resulting in path coordinate $s^i[m]$, velocity $v^i[\frac{m}{s}]$, acceleration $a^i[\frac{m}{s^2}]$ and relative distance $d = |s^2 - s^1|[m]$.

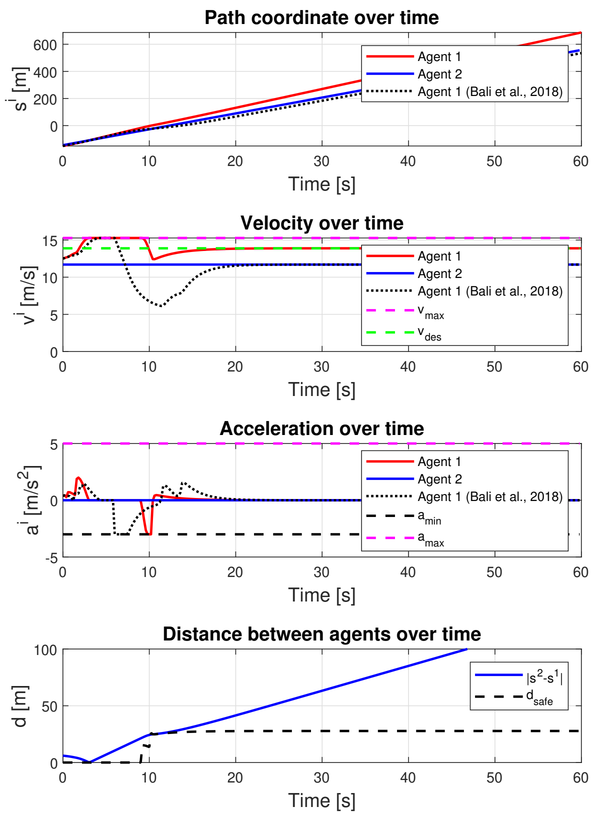

Fig. 4. Merging in front Agent 2 behavior, resulting in path coordinate $s^i[m]$, velocity $v^i[\frac{m}{s}]$, acceleration $a^i[\frac{m}{s^2}]$ and relative distance $d = |s^2 - s^1|[m]$.

accelerates towards the velocity of Agent 2 and merges behind Agent 2 with a safe distance. Agent 1 maintains the speed of Agent 2 at a proper distance in the target lane (note, that this is not the minimum safe distance, because this is not present in the minimization of the cost function). During the merging process the relative distance between the agents always satisfies the safe distance. Note that (Bali et al., 2018)-controller velocity shows a similar behavior, the differences between the controllers becomes visible in Scenario 2. The proposed merging MPC behaves properly, essentially integrating a merging controller and an ACC, always adhering to all (safety) constraints.

5.2 *Scenario 2: Agent 1 merges in front Agent 2*

In order to verify the behavior of Agent 1 when it merges in front of Agent 2, the initial velocities $v_0^1 = 12.5[\frac{m}{s}]$, $v_0^2 = 11.7[\frac{m}{s}]$ and $\Delta v_0 = v_0^2 - v_0^1[\frac{m}{s}]$ are selected. The MPC results are visualised in Fig. 4. These results are obtained in a maximum computation time of $0.5316[s]$ [1], what is larger then the sampling time of $T_s = 0.2[s]$, but in the same order of magnitude.

Agent 1 starts in the ego lane behind Agent 2. However, Agent 1's velocity is larger than the velocity of Agent 2. Therefore, the optimal solution is that Agent 1 accelerates even more, until it drives the maximum allowed velocity v_{max}. Agent 1 overtakes Agent 2 (with $\Delta s \geqslant 0$ in different lanes, i.e., no aggressive cut-in scenario). Thereafter, Agent 1 decelerates and maintains a constant velocity converging towards the desired velocity v_{ref}, hence it behaves as a cruise controller (CC). After the merge, Agent 1 drives further away from Agent 2. During the merging process the relative distance between the agents always satisfies the safe distance, as such showing that our MPC solution works as desired. Note that in the (Bali et al., 2018)-controller the velocity goes towards the maximum allowed velocity as well, but for a shorter amount of time. For this reason, the velocity of the ego vehicle is too low to merge in front of Agent 2 and it is decided to merge behind, in contrast to the merging in front of our proposed algorithm. In this scenario leading to better performance, due to the definition of a smaller terminal set in the (Bali et al., 2018)-controller. Note that, in other scenarios the (Bali et al., 2018)-controller, can merge in front as well. Interestingly, the overall proposed MPC scheme integrates merging, a CC and ACC functionality, as shown with the simulations.

6. CONCLUSIONS & FUTURE WORK

To overcome current challenges in trajectory generation for autonomous vehicles, this paper introduces a new MPC-MIQP algorithm for longitudinal lane merging automation. To extend the functionality and its generalisability, this controller is able to function during the lane merge and continues with (A)CC-functionality afterwards. We showed that the proposed algorithm is recursively feasible, using a larger controlled invariant terminal set than known in literature. Therefore, the control scheme adheres to the safety constraints during the complete merging process. Interestingly, the overall MPC scheme integrates merging, a CC and ACC functionality, as shown with the simulations. Future work will focus on considering different driving behavior of other road vehicles with uncertainty regarding their predicted behaviors (e.g., wider velocity profiles), improve scalability of the algorithm by considering multiple cars and consider lateral behavior of the ego vehicle. Also we strive for guarantees regarding safe robust recursive feasibility allowing variations in the velocity of the target vehicle and to evaluate this algorithm in real-world experiments.

REFERENCES

Alcalá, E., Puig, V., & Quevedo, J. (2019). LPV-MPC control for autonomous vehicles. IFAC-PapersOnLine, 52(28), 106–113.

Bali, C., & Richards, A. (2018). Merging vehicles at junctions using mixed-integer model predictive control. European Control Conference, pp. 1740-1745.

Bemporad, A., & Morari, M. (1999). Control of systems integrating logic, dynamics, and constraints. Automatica, 35(3), pp. 407–427.

Bouton, M., Nakhaei, A., Fujimura, K., & Kochenderfer, M. J. (2019). Cooperation-aware reinforcement learning for merging in dense traffic. IEEE Intelligent Transportation Systems Conference, pp 3441-3447.

Bouton, M., Nakhaei, A., Isele, D., Fujimura, K., & Kochenderfer, M. J. (2020). Reinforcement learning with iterative reasoning for merging in dense traffic. IEEE International Conference on Intelligent Transportation Systems, pp. 1-6.

Cao, W., Mukai, M., & Kawabe, T. (2019). Merging trajectory generation method using real-time optimization with enhanced robustness against sensor noise. Artificial Life and Robotics, 24(4), pp. 527–533.

Dixit, S., Montanaro, U., Fallah, S., Dianati, M., Oxtoby, D., Mizutani, T., & Mouzakitis, A. (2018). Trajectory planning for autonomous high-speed overtaking using MPC with terminal set constraints. 2018 21st International Conference on Intelligent Transportation Systems (ITSC).

Hubmann, C., Schulz, J., Xu, G., Althoff, D., & Stiller, C. (2018). A belief state planner for interactive merge maneuvers in congested traffic. International Conference on Intelligent Transportation Systems, pp. 1617-1624.

Ioan, D., Prodan, I., Olaru, S., Stoican, F., & Niculescu, S.-I. (2021). Mixed-integer programming in motion planning. Annual Reviews in Control, 51, pp. 65–87.

Katriniok, A., Kleibaum, P., & Joševski, M. (2017). Distributed model predictive control for intersection automation using a parallelized optimization approach. IFAC-PapersOnLine, 50(1), pp. 5940–5946.

Kerrigan, E. C., & Maciejowski, J. M. (2002). Invariant sets for constrained nonlinear discrete-time systems with application to feasibility in model predictive control. Proceedings of the 39th IEEE Conference on Decision and Control (Cat. No.00CH37187), pp. 4951-4956.

Khonji, M., Dias, J., Alyassi, R., Almaskari, F., & Seneviratne, L. (2020). A risk-aware architecture for autonomous vehicle operation under uncertainty. IEEE International Symposium on Safety, Security, and Rescue Robotics, pp. 311-317.

Mukai, M., Natori, H., & Fujita, M. (2017). Model predictive control with a mixed integer programming for merging path generation on motor way. IEEE Conference on Control Technology and Applications, pp. 2214-2219.

Nagahama, A., Suehiro, Y., Wada, T., & Sonoda, K. (2021). Assistance method for merging based on a probability regression model. IEEE Transactions on Intelligent Transportation Systems: A Publication of the IEEE Intelligent Transportation Systems Council, 22(5), pp. 2902–2912.

Naus, G. J. L., Ploeg, J., Van de Molengraft, M. J. G., Heemels, W. P. M. H., & Steinbuch, M. (2010). Design and implementation of parameterized adaptive cruise control: An explicit model predictive control approach. Control Engineering Practice, 18(8), pp. 882–892.

van Nunen, E., Verhaegh, J., Silvas, E., Semsar-Kazerooni, E., & van de Wouw, N. (2017). Robust model predictive cooperative adaptive cruise control subject to V2V impairments. IEEE International Conference on Intelligent Transportation Systems, pp. 1-8.

Paden, B., Cap, M., Yong, S. Z., Yershov, D., & Frazzoli, E. (2016). A survey of motion planning and control techniques for self-driving urban vehicles. IEEE Transactions on Intelligent Vehicles, 1(1), pp. 33–55.

Quirynen, R., Berntorp, K., Kambam, K., & Di Cairano, S. (2020). Integrated obstacle detection and avoidance in motion planning and predictive control of autonomous vehicles. American Control Conference, pp. 1203-1208.

Rawlings, J. B. (2009). Model predictive control: Theory and design. Nob Hill Pub.

Roelofsen, M. T. A. (2009). Safe lane changing A study into the practical implementation of the Lane Change Assistant. University of Twente.

J3016_201806: Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles - SAE international. (n.d.). Sae.org. Retrieved September 14, 2022, from https://www.sae.org/standards/content/j3016_201806/

Schweidel, K. S., Koehler, S. M., Desaraju, V. R., & Baric, M. (2022). Driver-in-the-loop contingency MPC with invariant sets. European Control Conference, pp. 808-813.

(N.d.-a). Swov.Nl. Retrieved September 14, 2022, from https://www.swov.nl/sites/default/files/publicaties/gearchiveerde-factsheet/uk/fs_acc_uk_archived.pdf

(N.d.-b). Swov.Nl. Retrieved September 14, 2022, from https://www.swov.nl/sites/default/files/publicaties/gearchiveerde-factsheet/uk/fs_headway_archived.pdf

Wang, H., Yuan, S., Guo, M., Li, X., & Lan, W. (2021). A deep reinforcement learning-based approach for autonomous driving in highway on-ramp merge. Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering, 235(10–11), pp. 2726–2739.

Wang, J., Zhang, L., Huang, Y., Zhao, J., & Bella, F. (2020). Safety of autonomous vehicles. Journal of Advanced Transportation, 2020, pp. 1–13.