

Interaction-Aware Trajectory Prediction and Planning in Dense Highway Traffic using Distributed Model Predictive Control

Erik Börve, Nikolce Murgovski, and Leo Laine

Abstract—In this paper we treat optimal trajectory planning for an autonomous vehicle (AV) operating in dense traffic, where vehicles closely interact with each other. To tackle this problem, we present a novel framework that couples trajectory prediction and planning in multi-agent environments, using distributed model predictive control. A demonstration of our framework is presented in simulation, employing a trajectory planner using non-linear model predictive control. We analyze performance and convergence of our framework, subject to different prediction errors. The results indicate that the obtained locally optimal solutions are improved, compared with decoupled prediction and planning.

I. INTRODUCTION

Considering the vast amount of existing traffic situations, trajectory planning in safety-critical traffic scenarios remains an active research topic. In this paper we consider one such scenario: highway driving maneuvers in *dense* traffic, where inter-vehicle distances can be smaller than the vehicles length. These maneuvers depend on complex driver-to-driver interactions, which pose challenges as the possibility for communication between vehicles is limited in practice [1]. Most commonly, drivers utilize the turn signal and adjust the relative distances to show intention. Further, maneuvers become additionally challenging when including heavy vehicle combinations (HVCs), due to e.g., limited visibility, and larger vehicle dimensions [2] (see, e.g., a forced lane change in Fig. 1).

Navigating in traffic can be considered as path planning with moving obstacles. Prior work has addressed this problem with Model Predictive Control (MPC), grid/tree-searching methods, and learning-based methods, see [3] and sources therein for a detailed review. The MPC approaches [4]–[6] have received attention due to the ability to enforce rigorous constraints on the dynamics of the own vehicle (henceforth referred to as the ego-vehicle) and safety with respect to avoiding collisions with other vehicles. In this context, MPC iteratively solves a constrained optimization problem based on current measurements of the environment and computes control actions for a finite time/space window, referred to as the horizon. However, without vehicle-to-vehicle communication, MPC requires a prediction of

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

Erik Börve and Nikolce Murgovski are with the Department of Electrical Engineering, Chalmers University of Technology, 412 96 Göteborg, Sweden {borerik, nikolce.murgovski}@chalmers.se

Leo Laine is with the Department of Mechanics and Maritime Science, Chalmers University of Technology, 412 96 Göteborg, Sweden leo.laine@chalmers.se

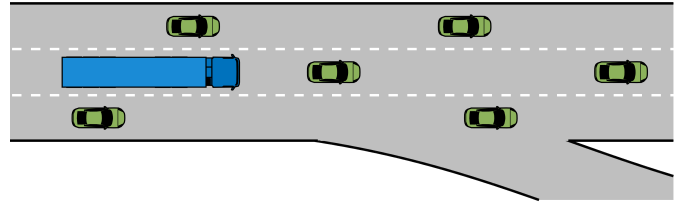


Fig. 1: Forced lane-change scenario for an HVC (blue) approaching an exit ramp on a multi-lane highway.

the surrounding traffic over the horizon [3]. Since humans (drivers) do not abide by Newtonian laws, predicting their trajectories is non-trivial and especially challenging when considering crowded environments with many interpersonal dependencies. The prediction of such trajectories has recently been addressed by learning-based predictors [7], which may also account for interactions among surrounding vehicles. The predicted trajectories can then be used to formulate collision avoidance constraints for an MPC-based trajectory planner, showing promising results in simulations of dense traffic scenarios [5]. In such interactive scenarios, there exists an inherent dependence between the planned MPC trajectory and the predicted trajectories of other road users. Hence, by decoupling the predictor and the planner, the predictions become less accurate and the MPC is not able to account for interactions with other vehicles. In current practical applications, interactive lane change maneuvers for AVs can become excessively defensive or even be avoided [2], [8].

To address this problem, we propose to couple the predictor with the planning controller. To the best of our knowledge, we propose a novel method based on Distributed Model Predictive Control (DMPC) and demonstrate the possibilities with a highway-autopilot controller for an HVC that considers interactions. The proposed method can in principle be combined with any predictor, possibly non-differentiable, that can be conditioned on a given trajectory, such as that proposed in [7]. Hence, our main contributions are as follows:

- 1) We propose a novel method for combining trajectory prediction and planning by utilizing DMPC.
- 2) We demonstrate our approach for a novel problem, employing non-linear MPC for an HVC in dense, and interaction-dependant traffic.
- 3) We provide an empirical analysis of the effect of prediction errors on the MPC performance and the convergence of our proposed method.

A simulation environment together with a model-based predictor was developed for the purpose of this paper.

II. DISTRIBUTED OPTIMAL CONTROL

MPC has been used in a wide array of multi-agent scenarios to optimize performance metrics and ensure safety-critical constraints. One implementation of MPC is a centralized approach, which considers a single controller that optimizes the combined objective of all agents over the horizon with respect to all states \mathbf{x} and control variables \mathbf{u} , most often subject to some constraints. In network control problems, agents with local objectives, local constraints, and coupled inequality constraints, are often encountered. Such problems can be solved centrally, but their computational complexity typically scales poorly with an increasing number of agents [9]. One approach to address this is to decompose the central problem using DMPC. The network control problem can instead be expressed for each agent i over a discrete horizon $k = 0, \dots, N$ as,

$$\begin{aligned} \min_{\mathbf{x}_i, \mathbf{u}_i} & f_{f,i}(\mathbf{x}_i(N)) + \sum_{k=0}^{N-1} f_i(\mathbf{x}_i(k), \mathbf{u}_i(k)) \\ \text{s.t. } & \mathbf{x}_i(k+1) = g_i(\mathbf{x}_i(k), \mathbf{u}_i(k)), \quad k = 0, \dots, N-1 \\ & h_i(\mathbf{x}_i(k), \mathbf{u}_i(k)) \leq 0, \quad k = 0, \dots, N-1 \\ & \mathbf{x}_i(0) = \mathbf{x}_{0,i} \end{aligned} \quad (1)$$

where $f_{f,i}$ and f_i describe the target and running cost of each agent, respectively, g_i describes the dynamics, h_i describes the coupled constraints, \mathbf{x}_i and \mathbf{u}_i are states and control inputs, respectively, and $\mathbf{x}_{0,i}$ are initial states. The coupling among the agents occurs in the inequality constraint h_i , which is a function of the states and control inputs of all the agents. A solution for the central problem is then obtained by overhead communication between all M agents. Rawlings et. al. [10] describes “Non-cooperative” DMPC, where communication is restricted to the solution of each agent’s respective optimization problem. In this setting, agents aim to minimize their own objective, treating other agents predicted trajectories $(\mathbf{x}_i, \mathbf{u}_i)^p$ as known disturbances. Each agent’s beliefs of the other agents’ trajectories are updated using a convex combination of their current optimal solution $(\mathbf{x}_i, \mathbf{u}_i)^*$ and the current iterate $(\mathbf{x}_i, \mathbf{u}_i)^p$ as,

$$\begin{aligned} \mathbf{x}_i^{p+1} & \leftarrow w_i \mathbf{x}_i^* + (1 - w_i) \mathbf{x}_i^p \\ \mathbf{u}_i^{p+1} & \leftarrow w_i \mathbf{u}_i^* + (1 - w_i) \mathbf{u}_i^p. \end{aligned} \quad (2)$$

The step size w_i represents each agent’s belief in their own current optimal solution where $0 < w_i < 1$. In practice, the convex step is iterated until convergence within a certain tolerance.

Do however note that for our problem we only control the ego-vehicle, while the trajectories of surrounding vehicles are estimated by a predictor. Considering non-Newtonian human drivers, the learned control policies of the surrounding vehicles could be non-differentiable, or even obtained by black-box approaches, and closed-form expression for their f_i , g_i or h_i functions may not be available. Yet we show in this paper that it may still be possible to implement a DMPC strategy using a gradient-based method for the ego-vehicle controller.

III. VEHICLE MODELLING

This section provides a model of the ego vehicle and the surrounding traffic used in simulations.

A. Ego-Vehicle Model

The ego-vehicle motion is modeled using a kinematic bicycle model, extended with a trailer, as

$$\dot{\mathbf{x}}_e = \begin{bmatrix} \dot{p}_x \\ \dot{p}_y \\ \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} = \begin{bmatrix} v_x \\ v_x \tan \theta_1 \\ a_v \cos \theta_1 \\ v_x \frac{\tan \delta}{\ell_1 \cos \theta_1} \\ v_x \frac{\sin(\theta_1 - \theta_2)}{\ell_2 \cos \theta_1} \end{bmatrix}, \quad \mathbf{u}_e = \begin{bmatrix} \delta \\ a_v \end{bmatrix}. \quad (3)$$

Here, \mathbf{x}_e are states that include x - and y -position at the tractor-trailer joint in the inertial frame p_x, p_y , longitudinal velocity at the tractor-trailer joint in the inertial frame v_x and the tractor and trailer angles with respect to the inertial frame θ_1, θ_2 . The vehicle states are controlled by the steering angle δ and the longitudinal acceleration a_v at the tractor-trailer joint expressed in the body frame. The wheelbase of the tractor and trailer are represented by ℓ_1, ℓ_2 , respectively. For simulation and optimization, the model is discretized as,

$$\mathbf{x}_e(k+1) = g_e(\mathbf{x}_e(k), \mathbf{u}_e(k)). \quad (4)$$

B. Traffic Model

The surrounding vehicles are modeled using a kinematic bicycle model. The state \mathbf{x}_i and control vector \mathbf{u}_i of a vehicle i are described as for the ego-vehicle but excluding the trailer state θ_2 . The longitudinal acceleration of each vehicle a_i is calculated based on tracking its own reference velocity and maintaining a safe distance to leading vehicles in the same lane. Additionally, vehicles can consider accelerating or decelerating to allow a vehicle in an adjacent lane to merge into their own, the extent of which is determined by how cooperative the vehicle is [11]. Such a traffic model can be described with a general function as,

$$\mathbf{x}_i(k+1) = g_i(\mathbf{x}(k), a_i(k), \phi) \quad (5)$$

where ϕ are model parameters and \mathbf{x} are the states of all surrounding vehicles. The steering angle δ of surrounding vehicles has been considered zero in the studied traffic scenarios, as focus is placed on whether surrounding vehicles can yield a sufficient gap for the ego vehicle to fit into by accelerating or decelerating. However, it is straightforward to apply the proposed method to scenarios where surrounding vehicle may also change lanes.

IV. EGO-VEHICLE PROBLEM FORMULATION

The trajectory planning problem, even for a single vehicle, is a mixed-integer program due to the need of choosing a target lane which is inherently an integer decision. Instead of solving a single mixed-integer problem, it was proposed in [4], [6] to solve three MPCs, each with a different target lane, i.e., keep current lane (no lane change), change to the left lane, and change to the right lane, noted as $j \in \{\text{nc}, \text{lc}, \text{rc}\}$, respectively. A decision manager compares the costs of the

optimal MPC solutions and returns the safe optimal choice, e.g., driving forward, and a desired optimal choice, e.g., a forced lane change.

Compared to [4], [6], we additionally consider a scenario with blocking traffic where the ego-vehicle is not able to immediately initiate a lane change, but it instead cautiously approaches adjacent vehicles to gauge their cooperativeness.

A. Collision Avoidance Constraints

As the collision avoidance constraints are based on estimations of trajectories ($\hat{\mathbf{x}}_i^j, \hat{\mathbf{u}}_i^j$) of surrounding vehicle i for each controller j , deviations from the true trajectories can yield infeasible problems in edge cases. Hence, we introduce slack variables $\lambda_e^j \in \mathbb{R}^{N \times M_e^j}$ to relax the collision avoidance constraints, where \mathbb{M}_v^j represents a subset of the surrounding vehicles, with size M_e^j . These receive a substantial penalty in the objective to remove the possibility of collisions in a practical sense but persistent feasibility is no longer rigorously established.

For the lane keeping controller $j = \text{nc}$, avoiding collisions amounts to maintaining a longitudinal distance to a leading vehicle i , while remaining in the current lane. The minimum allowed longitudinal distance has to satisfy

$$c_e^{\text{nc}}(\mathbf{x}_e^{\text{nc}}, \hat{\mathbf{x}}_i^{\text{nc}}) = p_{x,e}^{\text{nc}} - \hat{p}_{x,i}^{\text{nc}} + \ell_1 + d_s + T_s v_{x,e}^{\text{nc}} + \lambda_{e,i}^{\text{nc}} < 0 \quad (6)$$

where d_s represents a distance margin, ℓ_1 is the tractor length, T_s represents the time-headway and $\lambda_{e,i}^{\text{nc}}$ is the slack variable. To avoid collisions with vehicles in adjacent lanes, the lateral position is constrained to the current lane margins, accounting for the maximum lateral span of the ego-vehicle $d_{w,e}$. A visualization of the constraints with the referenced parameters is displayed in Fig. 2a.

In the lane changing case, $j \in \{\text{lc}, \text{rc}\}$, we aim to constrain both the longitudinal and lateral distance to some vehicle i . This yields a possibly non-differentiable constraint which can be formulated as a mixed integer problem (MIP). However, as MIPs are significantly more computationally intensive to solve, we employ a smooth approximation \tilde{y}_i^j of the constraint boundaries using tanh-functions, as

$$\tilde{y}_i^j(\mathbf{x}_e^j, \hat{\mathbf{x}}_i^j) = \frac{\alpha_{0,i}^j(\hat{p}_{y,i}^j)}{2} \left(\tanh(p_{x,e}^j - \hat{p}_{x,i}^j + \alpha_{1,i}^j) + \tanh(\hat{p}_{x,i}^j - p_{x,e}^j + \alpha_{2,i}^j) \right) + \alpha_{3,i}^j(p_{y,e}^j, \hat{p}_{y,i}^j). \quad (7)$$

Here, $\alpha_{0,i}^j$ and $\alpha_{1,i}^j$ scale the constraint in lateral and longitudinal distance, respectively, and incorporate the surrounding vehicle position, width and length, together with those of the ego-vehicle, plus a safety margin. Lastly, $\alpha_{3,i}^j$ shifts the constraint in lateral distance based on the current lateral position of the surrounding and ego-vehicle. A visualization of the constraints for both left and right lane changes are displayed in Fig. 2b and Fig. 2c. Hence, the collision avoidance constraints can be formulated as,

$$c_e^j(\mathbf{x}_e^j, \hat{\mathbf{x}}_i^j) = \beta_i^j(\tilde{y}_i^j(\mathbf{x}_e^j, \hat{\mathbf{x}}_i^j) + d_{w,e} - p_{y,e}^j + \lambda_{e,i}^j) < 0 \quad (8)$$

where $\beta_i^j \in \{-1, 1\}$ converts between an upper and lower constraint on lateral position, depending on the vehicle i and

controller j . This constraint is repeated for all surrounding vehicles $i \in \mathbb{M}_v^j$.

Finally, constraints (6) and (8) together with the physical limitations on the ego-vehicle states and control input form the general constraints for each controller j ,

$$h_e^j(\mathbf{x}_e^j(k), \mathbf{u}_e^j(k), \lambda_e^j(k), \hat{\mathbf{x}}^j(k), \hat{\mathbf{u}}^j(k)) \leq 0, \quad (9)$$

where $\hat{\mathbf{x}}^j(k)$ and $\hat{\mathbf{u}}^j(k)$ gather the estimated/predicted states and control inputs of all surrounding vehicles.

B. Optimal Control Problem Formulation

Each MPC j of the ego-vehicle aims to track a reference in longitudinal velocity and lateral position while minimizing the control input. The objective also addresses driver comfort by minimizing the change in acceleration and steering angle. Expressing the objective over the horizon for each controller j , including slack variables, yields,

$$f_e^j(\mathbf{X}_e^j, \mathbf{U}_e^j, \Lambda_e^j) = \|\mathbf{x}_e^j(N) - \mathbf{x}_{e,r}^j(N)\|_P^2 + \|\lambda_e^j(N)\|_{Q_s}^2 \quad (10a)$$

$$+ \sum_{k=0}^{N-1} \|\mathbf{x}_e^j(k) - \mathbf{x}_{e,r}^j(k)\|_Q^2 + \|\lambda_e^j(k)\|_{Q_s}^2 \quad (10b)$$

$$+ \sum_{k=0}^{N-1} \|\mathbf{u}_e^j(k) - \mathbf{u}_{e,r}^j(k)\|_R^2 \quad (10c)$$

$$+ \sum_{k=0}^{N-2} \|\mathbf{u}_e^j(k+1) - \mathbf{u}_e^j(k)\|_{R_d}^2 \quad (10d)$$

where $Q_s = q_s I_{M_e}$, $R = \text{diag}(q_a, q_\delta)$, $R_d = \text{diag}(q_{da}, q_{d\delta})$ and $Q = \text{diag}(0, q_y, q_v, 0, 0)$ are positive semi-definite matrices and $\mathbf{x}_{e,r}^j = [0, p_{y,r}^j, v_{x,r}^j, 0, 0]^T$, $\mathbf{u}_{e,r}^j = [0, 0]^T$ represent the state and input reference for each controller. The ego vehicle states over the horizon are gathered in

$$\mathbf{X}_e^j = \{\mathbf{x}_e^j(0), \mathbf{x}_e^j(1), \dots, \mathbf{x}_e^j(N)\}, \quad j \in \{\text{nc}, \text{lc}, \text{rc}\} \quad (11)$$

and similarly are the control inputs and slack variables in \mathbf{U}_e^j and Λ_e^j . The positive definite terminal cost matrix P is determined using an infinite horizon linear quadratic regulator (LQR) by solving the discrete time Riccati differential equation for linearized dynamics (3) around $\mathbf{x}_{e,r}$ and $\mathbf{u}_{e,r}$, see [10] for more details.

We can now express the optimal control problem for the ego-vehicle and each respective controller $j \in \{\text{nc}, \text{lc}, \text{rc}\}$,

$$\min_{\mathbf{X}_e^j, \mathbf{U}_e^j, \Lambda_e^j} f_e^j(\mathbf{X}_e^j, \mathbf{U}_e^j, \Lambda_e^j) \quad (12a)$$

$$\text{s.t. } \mathbf{x}_e^j(k+1) = g_e(\mathbf{x}_e^j(k), \mathbf{u}_e^j(k)) \quad (12b)$$

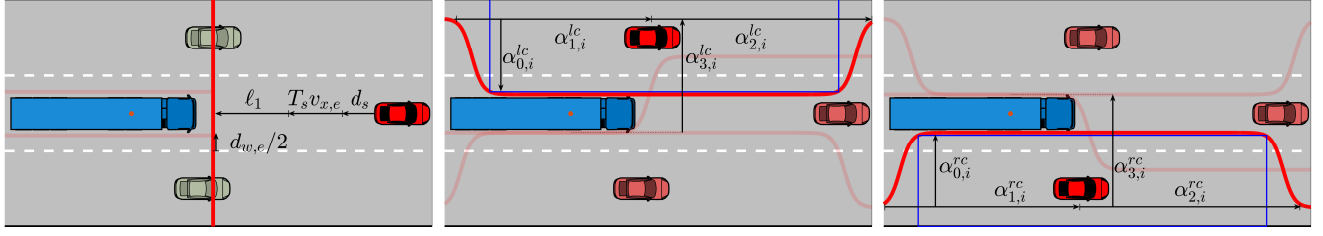
$$h_e^j(\mathbf{x}_e^j(k), \mathbf{u}_e^j(k), \lambda_e^j(k), \hat{\mathbf{x}}^j(k), \hat{\mathbf{u}}^j(k)) \leq 0 \quad (12c)$$

$$\lambda_e^j(k) \leq 0 \quad (12d)$$

$$\mathbf{x}_e^j(0) = \mathbf{x}_{0,e} \quad (12e)$$

$$(\hat{\mathbf{X}}^j, \hat{\mathbf{U}}^j) = \Pi(\cdot, \mathbf{X}_e^j, \mathbf{U}_e^j) \quad (12f)$$

where (12b)–(12d) are imposed $\forall k = 0, \dots, N-1$ and $(\hat{\mathbf{X}}^j, \hat{\mathbf{U}}^j)$ are estimated/predicted trajectories of surrounding vehicles over the horizon, gathering $\hat{\mathbf{x}}^j(k)$ and $\hat{\mathbf{u}}^j(k)$, $\forall k$ similarly as in (11). The predictor Π will be described in more details in Section V. It is important to note here that sensitivities (derivatives) of the predictor Π may not be available and problem (12) may not be possible to directly



(a) Collision avoidance constraints for the lane keeping controller. (b) Collision avoidance constraints for the left lane change controller. (c) Collision avoidance constraints for the right lane change controller.

Fig. 2: Collision avoidance constraints for each controller. Red vehicles are explicitly included in constraints, indicated metrics refer to the opaque vehicles and constraints. The constrained point on the ego-vehicle is indicated with a red dot, (b) and (c) additionally display the approximated discontinuous constraint.

cast to a standard nonlinear program (NLP). Indeed, the goal of this paper is to decouple (12f) such that the subproblem in (12) can be cast as a standard NLP.

After each of the three MPCs are finished computing, the decision manager selects the control signals from the MPC that optimizes,

$$\arg \min_{\mathbf{x}_e^j, \mathbf{U}_e^j} q_e f_e^j(\cdot) + q_c f_c^j + q_s f_s^j(\mathbf{x}_{0,e}), \quad j \in \{\text{nc}, \text{lc}, \text{rc}\} \quad (13)$$

where q_e, q_c, q_s are positive scaling parameters and f_e^j is current cost of the j^{th} MPC. The function f_c^j introduces a cost on switching j based on m previous decisions as,

$$f_c^j = \sum_{i=1}^m \mathbf{I}(H(i) - j) \quad (14)$$

where H is the history of previous choices of j and \mathbf{I} is an indicator function. The function f_s^j introduces a cost on the distance to a highway exit as,

$$f_s^j(\mathbf{x}_{0,e}) = \left(1 - \left(\frac{d_{\text{exit}} - p_{0,x}}{d_{\text{max}}}\right)^\gamma\right) \mathbf{I}(j^* - j) \quad (15)$$

where d_{exit} is the distance to the exit, $d_{\text{max}} > d_{\text{exit}}$, $\gamma \in [0, 1]$ are considered tuning parameters and j^* is the decision that guides the ego-vehicle towards the highway exit [6].

V. PREDICTING AND PLANNING USING DMPC

As outlined in Section IV, a trajectory planner in highway traffic can be designed using MPC with local objectives and constraints, combined with coupled inequality constraints. Without vehicle-to-vehicle communication, the coupled constraints require a prediction of the surrounding vehicles trajectories, which in turn depend on the planned ego-vehicle trajectories. Consider a general predictor Π as in [7], that returns trajectories $(\hat{\mathbf{X}}, \hat{\mathbf{U}})$ over a future horizon. The predictor utilizes past observations of all vehicles $\mathbf{X}_{\text{obs}}, \mathbf{U}_{\text{obs}}$ together with the planned trajectory of the ego-vehicle $\mathbf{X}_e, \mathbf{U}_e$ over the future horizon as,

$$(\hat{\mathbf{X}}, \hat{\mathbf{U}}) = \Pi(\mathbf{X}_{\text{obs}}, \mathbf{U}_{\text{obs}}, \mathbf{X}_e, \mathbf{U}_e). \quad (16)$$

In the DMPC formalism, the predicted variables can be treated as an approximation of the local solution of each agent's optimal control problem (1). Hence, if the prediction $(\hat{\mathbf{X}}^p, \hat{\mathbf{U}}^p)$ at some iterate p can be used to express the coupled constraints, it is possible to apply the method presented

in Section II to formulate an iterative algorithm that attempts to solve the centralized problem by distributing the ego-vehicle planner from its predictor. Given a fixed prediction $(\hat{\mathbf{X}}^p, \hat{\mathbf{U}}^p)$, problem (12) becomes a smooth NLP as,

$$P_{\text{MPC}}(\hat{\mathbf{X}}^p, \hat{\mathbf{U}}^p) = \arg \min_{\mathbf{X}_e, \mathbf{U}_e, \Lambda_e} f_e(\mathbf{X}_e, \mathbf{U}_e, \Lambda_e) \quad (17a)$$

$$\text{s.t. (12b) - (12e)} \quad (17b)$$

here, and onward omitting the index j for convenience. Predicted and planned trajectories that remain constant over iterates p indicate that both concur. Note that the convergence that has been guaranteed for linear systems in [10] cannot be generalized to this problem as we consider non-linear MPC with prediction errors and a general predictor. To prevent divergence, we propose a loss metric L , which can treat prediction and planning jointly,

$$L^{p+1} = \|\hat{\mathbf{X}}^{p+1} - \hat{\mathbf{X}}^p\|_2 + \|\hat{\mathbf{U}}^{p+1} - \hat{\mathbf{U}}^p\|_2 + \|\mathbf{X}_e^{p+1} - \mathbf{X}_e^p\|_2 + \|\mathbf{U}_e^{p+1} - \mathbf{U}_e^p\|_2. \quad (18)$$

Our approach enforces the loss metric to decrease at each iteration, i.e. $L^{p+1} < L^p$. If the loss metric stops decreasing, or if it drops below a tolerance ϵ , the iteration is terminated. This trivially means that the loss metric cannot diverge from the closest local solution of the trajectory initialization.

Algorithm 1 displays the formulation of the coupled predicting and planning method, employing the DMPC scheme for (17). In a proper MPC setting, we introduce an index κ when MPC is updated, such that each variable, e.g., the ego vehicle state, can be written as $\mathbf{x}_e(\kappa + k|\kappa)$, or even simpler as $\mathbf{x}_e(k|\kappa)$. The predicted trajectories over the horizon, e.g., the states as in (11), can be written as $\mathbf{X}_{e,\kappa}^p$. In line 4, the algorithm computes an optimal MPC solution, denoted as $(\mathbf{X}_{e,\kappa}^{p+1}, \mathbf{U}_{e,\kappa}^{p+1})^*$, by iterating the DMPC method over p towards p_{max} at each time step κ until κ_{end} . Notice that the ego trajectories updated in line 5 are used only for computing the loss metric and by the predictor, while the returned DMPC solution $(\mathbf{X}_{e,\kappa}^*, \mathbf{U}_{e,\kappa}^*)$ is from one of the iterates in line 4, i.e., before performing the update.

To derive a good initial guess for variables at time step κ , we define a function $\text{shift}(\cdot)$ that shifts the optimal variables from time step $\kappa - 1$ and pads the last value using zero control [10].

Algorithm 1 Predicting and Planning using DMPC

```

1: for  $\kappa = 1, \dots, \kappa_{\text{end}}$  do
2:   Initialize:  $(\mathbf{X}_{e,\kappa}^0, \mathbf{U}_{e,\kappa}^0) = \text{shift}(\mathbf{X}_{e,\kappa-1}^*, \mathbf{U}_{e,\kappa-1}^*)$ 
                $(\hat{\mathbf{X}}_{\kappa}^0, \hat{\mathbf{U}}_{\kappa}^0) = \Pi(\mathbf{X}_{\text{obs},\kappa}, \mathbf{U}_{\text{obs},\kappa}, \mathbf{X}_{e,\kappa}^0, \mathbf{U}_{e,\kappa}^0)$ 
                $L^0 = \infty$ 
3:   for  $p = 0, \dots, p_{\text{max}}$  do
4:     Plan:  $(\mathbf{X}_{e,\kappa}^{p+1}, \mathbf{U}_{e,\kappa}^{p+1})^* = P_{\text{MPC}}(\hat{\mathbf{X}}_{\kappa}^p, \hat{\mathbf{U}}_{\kappa}^p)$ 
5:     Update:  $\mathbf{X}_{e,\kappa}^{p+1} \leftarrow w_e(\mathbf{X}_{e,\kappa}^{p+1})^* + (1 - w_e)\mathbf{X}_{e,\kappa}^p$ 
               $\mathbf{U}_{e,\kappa}^{p+1} \leftarrow w_e(\mathbf{U}_{e,\kappa}^{p+1})^* + (1 - w_e)\mathbf{U}_{e,\kappa}^p$ 
6:     Predict:  $(\hat{\mathbf{X}}_{\kappa}^{p+1}, \hat{\mathbf{U}}_{\kappa}^{p+1}) = \Pi(\mathbf{X}_{\text{obs},\kappa}, \mathbf{U}_{\text{obs},\kappa}, \mathbf{X}_{e,\kappa}^{p+1}, \mathbf{U}_{e,\kappa}^{p+1})$ 
7:     Update:  $\hat{\mathbf{X}}_{\kappa}^{p+1} \leftarrow w\hat{\mathbf{X}}_{\kappa}^p + (1 - w)\hat{\mathbf{X}}_{\kappa}^p$ 
               $\hat{\mathbf{U}}_{\kappa}^{p+1} \leftarrow w\hat{\mathbf{U}}_{\kappa}^p + (1 - w)\hat{\mathbf{U}}_{\kappa}^p$ 
8:     Compute:  $L^{p+1}$  using (18)
9:     if  $L^p < L^{p+1}$  then
10:        $(\mathbf{X}_{e,\kappa}^*, \mathbf{U}_{e,\kappa}^*) = (\mathbf{X}_{e,\kappa}^p, \mathbf{U}_{e,\kappa}^p)^*$ , exit the loop
11:     else if  $L^{p+1} < \epsilon$  then
12:        $(\mathbf{X}_{e,\kappa}^*, \mathbf{U}_{e,\kappa}^*) = (\mathbf{X}_{e,\kappa}^{p+1}, \mathbf{U}_{e,\kappa}^{p+1})^*$ , exit the loop
13:     end if
14:   end for
15:   Apply the first control action  $u_e(0|\kappa)$  from  $\mathbf{U}_{e,\kappa}^*$ 
16: end for

```

VI. SIMULATION STUDY

This section provides a case study with the proposed methods, investigating performance and convergence with respect to prediction errors.

A. Simulation Setup

The simulation environment¹ controls the surrounding vehicles using the models described in Section III-B and the ego-vehicle uses our MPC. The traffic model parameters are randomized using uniform distributions as $\phi = \phi_{\text{mean}} + \mathcal{U}(\phi_{\text{min}}, \phi_{\text{max}})$. The MPC problem is formulated using *Casadi*, running the *IpOpt* solver [12]. We demonstrate the approach presented in Section V using a model-based predictor Π , utilizing the model in Section III-B with added noise $\mathcal{N}(0, \sigma_a)$ to the control inputs of the surrounding vehicles.

The predictor hence utilize a measurement of surrounding vehicles, repeatedly applies the ground-truth model using the planned ego-vehicle trajectory and adds normal distributed noise to introduce a prediction error that can be scaled with σ_a . Note that this predictor will be less ideal in estimating and predicting the motion of real vehicles, which may require learning-based methods. Yet, we deem this approach suitable to clearly demonstrate the proposed DMPC algorithm and investigate the impact of prediction errors.

B. Test Scenarios

The dense traffic environment is simulated by randomly sampling M vehicles in close proximity of the ego-vehicle located in the middle lane. Longitudinal inter-vehicle distances in adjacent lanes are kept smaller than the length of the ego-vehicle. The simulations are initialized such that a lane change is feasible, if properly handled by the controller.

¹Code: <https://github.com/BorveErik/Autonomous-Truck-Sim>

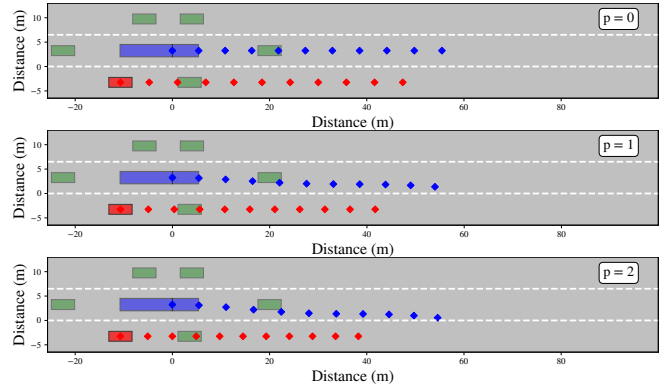


Fig. 3: Example of a generated FLC scenario. Each plot displays the state trajectory of two vehicles in corresponding color at iteration p .

Fig. 3 illustrates a sampled scenario. In this setting, we consider a “Forced lane change” maneuver (FLC) where the ego-vehicle is required to reach a highway exit at $p_{x,\text{exit}} = 250\text{m}$, ahead in the rightmost lane. The desired speed $v_{x,r}$ is 30 km h^{-1} and a lane change is defined as successful if the right lane’s center $p_{y,r}^c$ is reached before the exit. The maximum considered time frame is $t_{\text{max}} = 30\text{ s}$. We study two separate controller architectures for the ego-vehicle: 1.) “DC-MPC”: The ego-vehicle considers decoupled prediction and planning; 2.) “PP-DMPC” The ego-vehicle couples prediction and planning using Algorithm 1. The PP-DMPC hyperparameters are chosen as: $p_{\text{max}} = 15$, $w = w_e = 1/(M + 1)$ and $\epsilon = 5$. Hence, both controllers utilize the same predictive capabilities of Π and gauge interactions with other vehicles by first approaching the lane markers of the desired lane. The PP-DMPC controller further considers how the ego-vehicle trajectory interacts with the predicted trajectory of the surrounding vehicles through Algorithm 1. Simulations are evaluated with three different noise settings for Π : $\sigma_{a,1} = 0.1$, $\sigma_{a,2} = 0.5$, and $\sigma_{a,3} = 1.0$. Note that the acceleration of each vehicle is confined within a defined physical limitation of $\pm 4\text{ m s}^{-2}$, both in simulation and in the predictor.

C. MPC Performance Results

1) *Qualitative Analysis*: Fig. 3 highlights the key characteristics of the coupled trajectory prediction and planning proposed in our method. The top plot displays a naive, but feasible initialization of the state trajectories for the ego-vehicle and an adjacent vehicle. As iterations in the distributed scheme continue, the predictor and planner reason via coupled variables. The middle plot displays emerging interactions between the planned ego-vehicle trajectory (blue) and an adjacent vehicle (purple) at $p = 5$. Finally, the bottom plot displays a converged PP-DMPC solution at $p = 15$, where the adjacent vehicle in the right lane have slowed down to let the ego vehicle make a lane change.

2) *Quantitative Analysis*: To evaluate the performance of both controllers, 100 scenarios were sampled. The overall performance is evaluated using the following: 1.) “Success Rate”: The percentage of scenarios that achieve the goal of the respective case; 2.) “Collision rate”: The percentage of scenarios resulting in a collision; 3.) “Time”: The average

TABLE I: Statistic results for all 100 sampled scenarios and each predictor as $(\sigma_{a,1}/\sigma_{a,2}/\sigma_{a,3})$. Relative controller cost are presented as a percentage.

Controller	Success (%)	Collision (%)	Time (s)	Total Cost (%)	Avg. Nr. of Iterations (p)	Convergence Rate (%)
DC-MPC	100/99/98	0/1/2	22.1/22.1/22.1	97.3/98.1/100.0	—	—
PP-DMPC	100/100/99	0/0/1	21.1/21.2/21.5	66.0/71.8/74.1	2.41/2.68/2.38	87.0/64.1/17.4

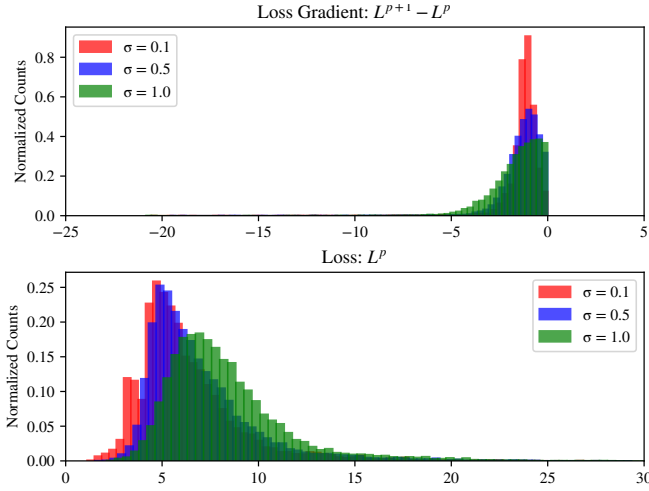


Fig. 4: Distribution of the loss (bottom) and its gradient (top).

completion time for the successful scenarios. We further treat “optimality” by considering: “Total Cost” as an evaluation of the objective (10) for all simulated trajectories. These metrics are presented in Table I where results for different predictors are indicated as $\sigma_{a,1}/\sigma_{a,2}/\sigma_{a,3}$. The PP-DMPC out-performs the DC-MPC in the investigated metrics, for all different noise levels. The performance of both the DC-MPC and the PP-DMPC decreases with an increasing σ_a . The relative improvements of the PP-DMPC over the DC-MPC also decreases with an increasing σ_a . This indicates that Algorithm 1, applied for our non-linear MPC can have benefits, however also that the extent of which is reliant on the predictors accuracy.

D. Convergence and Prediction Errors

The average number of iterations for the PP-DMPC and the convergence rate for each predictor is displayed in Table I. This indicates that the convergence of Algorithm 1 is dependent on the accuracy of the predictor. The distribution of the loss L and its gradient $L^{p+1} - L^p$, is displayed in Fig. 4. On average, larger prediction errors result in a larger loss and larger corresponding gradients. The loss gradients are indeed never positive which empirically shows that the algorithm does not diverge from the closest local minima.

VII. CONCLUSIONS AND FUTURE WORK

The largest challenge for our proposed method is related to real-world applicability. Based on our simulation study, we hope to investigate if learning-based predictors, trained on real-world traffic data, can obtain a sufficient accuracy for our proposed method. Another challenge relates to the computational complexity. We hope to investigate this in future work, for example by considering a sub-optimal MPC formulation. Our constraint on a continual decreasing loss is also conservative and might limit exploration of the

non-linear state space. A more rigorous study of convergence might reveal more effective criteria and ideal hyperparameter choices, which could improve MPC performance and the convergence rate. Lastly, as the collision avoidance constraints need to be relaxed to aid feasibility, the planned trajectory can result in collisions in edge cases. This could be further mitigated by several approaches e.g., considering robust MPC [3] with stochastic predictors [7], and identifying and aborting dangerous maneuvers [2].

ACKNOWLEDGMENT

The authors would like to thank Morteza Haghir Chehreghani, Deepthi Pathare and Stefan Börjesson for our valuable discussions.

REFERENCES

- [1] P. Hidas, “Modelling vehicle interactions in microscopic simulation of merging and weaving,” *Transportation Research Part C: Emerging Technologies*, vol. 13, no. 1, pp. 37–62, 2005.
- [2] P. Nilsson, L. Laine, and B. Jacobson, “A simulator study comparing characteristics of manual and automated driving during lane changes of long combination vehicles,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 9, pp. 2514–2524, 2017.
- [3] S. Dixit, S. Fallah, U. Montanaro, M. Dianati, A. Stevens, F. McCullough, and A. Mouzakitis, “Trajectory planning and tracking for autonomous overtaking: State-of-the-art and future prospects,” *Annual Reviews in Control*, vol. 45, pp. 76–86, 2018.
- [4] N. Murgovski and J. Sjöberg, “Predictive cruise control with autonomous overtaking,” in *2015 54th IEEE Conference on Decision and Control (CDC)*. IEEE, 2015, pp. 644–649.
- [5] S. Bae, D. Isele, A. Nakhaei, P. Xu, A. M. Añon, C. Choi, K. Fujimura, and S. Moura, “Lane-change in dense traffic with model predictive control and neural networks,” *IEEE Transactions on Control Systems Technology*, 2022.
- [6] J. Karlsson, N. Murgovski, and J. Sjöberg, “Optimal trajectory planning and decision making in lane change maneuvers near a highway exit,” in *2019 18th European Control Conference (ECC)*. IEEE, 2019, pp. 3254–3260.
- [7] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, “Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data,” in *Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII* 16. Springer, 2020, pp. 683–700.
- [8] D. Sadigh, S. Sastry, S. A. Seshia, and A. D. Dragan, “Planning for autonomous cars that leverage effects on human actions,” in *Robotics: Science and systems*, vol. 2. Ann Arbor, MI, USA, 2016, pp. 1–9.
- [9] F. Bertilsson, M. Gordon, J. Hansson, D. Möller, D. Söderberg, Z. Zhang, and K. Åkesson, “Centralized versus distributed nonlinear model predictive control for online robot fleet trajectory planning,” in *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2022, pp. 701–706.
- [10] J. B. Rawlings, D. Q. Mayne, and M. Diehl, *Model predictive control: theory, computation, and design*. Nob Hill Publishing Madison, WI, 2017, vol. 2, ch. 6, pp. 369–450.
- [11] B. Brito, A. Agarwal, and J. Alonso-Mora, “Learning interaction-aware guidance for trajectory optimization in dense traffic scenarios,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 10, pp. 18 808–18 821, 2022.
- [12] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, “Casadi: a software framework for nonlinear optimization and optimal control,” *Mathematical Programming Computation*, vol. 11, pp. 1–36, 2019.