Lane change in automated driving: an explicit coordination strategy

Alessandro Falsone, *Member, IEEE*, Beatrice Melani, and Maria Prandini, *Fellow, IEEE*

***Abstract*— We address a multi-vehicle automated driving scenario, where a vehicle has to change lane and merge in a platoon in a one-way roadway with two lanes. We focus on the coordination phase of the lane change, where vehicles in the platoon need to create a gap for the merging vehicle to enter safely following a pre-computed optimal trajectory. The goal is pre-computing also the multi-vehicle coordination strategy, so as to limit the computational and communication effort involved in its online implementation. This is achieved by considering the platoon as if it was composed of an infinite number of vehicles and solving a multi-parametric optimization program providing the coordination strategy as an explicit function of position and velocity of the ego vehicle, integrating a multi-class classifier to identify the best merging position. Numerical simulations show that the resulting performance degradation when implementing the strategy on a finite platoon is limited to boundary effects at its head and tail.**

***Index Terms*— Automated driving, explicit strategy, multi-agent coordination.**

I. Introduction

AUTOMATED driving is expected to provide various benefits not only to a vehicle user, but also to transportation systems operators and the whole society, as discussed in [1]. In particular, the introduction of automated driving systems will reduce the occurrence of accidents, cut down driving times and, consequently, emissions and energy consumption.

Lane change is a complex maneuver that can prove to be challenging even for a human driver since it requires particular attention to the surroundings and the ability to assess distances and speeds to avoid collisions, making it a frequent cause of accidents [2]. As such, it is considered as pivotal when addressing automated driving.

Different strategies have been devised to perform an automated lane change maneuver. The strategy can be cooperative if vehicles have a means to communicate with each other, otherwise vehicles have to design their own trajectory based on exogenous information gathered from the environment. Lane change maneuver design for connected automated vehicles is discussed in the survey paper [3], together with enabling technologies and control architectures. Trajectories of vehicles are typically jointly optimized (cf. [4], [5]), possibly solving a nonlinear optimal control problem, [6]. A distinguishing feature of [5] is that lane change on a three-lane one-way road is studied, although cooperation between two vehicles only is considered. Lane change problems with many cooperative vehicles are indeed more complex and contributions in the literature are limited (see [4] and the references therein).

A model predictive control framework is adopted in [7] for the non-cooperative case, assuming the future behavior of surroundings vehicles to be known. Scenarios where the other vehicles behavior is considered unpredictable are also explored, albeit not extensively, in [8]. The aforementioned works makes use of different approaches (e.g., parametric curves [5], [8], decoupling of the longitudinal motion from the lateral one [7], a single-track kinematic model [6], optimization of a simple double-integrator model [4]) to compute the vehicle trajectory. In those works where a nonlinear kinematic model is considered, the solution relies on the pure rolling assumption on the tires, which holds at low speed, but it is inadequate to represent faster maneuvers.

In [9], a multi-vehicle lane change maneuver is considered, where a vehicle has to merge into a platoon moving on the adjacent lane in a two-lane one-way roadway. To handle complexity, the merging maneuver is split into two phases: i) a coordination phase, where all vehicles in the platoon keep traveling on their lane while cooperating to create (in minimum time) a gap for the ego vehicle to enter safely, and ii) a merging phase, where vehicles in the platoon travel at constant speed along their lane while the ego vehicle performs a (optimal) merging maneuver. The first phase is formulated as a convex optimization problem, which has to be solved online $m+1$ times if the platoon is composed of m vehicles, so as to find the best merging position (platoon head/tail or between two subsequent vehicles) for the ego vehicle. During this phase a simple double-integrator model of their longitudinal dynamics can be used. All vehicles have to reach a suitable relative longitudinal position and reference speed at the end of the first phase. This serves as a starting point for the second phase, which is formulated as an optimal control problem involving only the ego vehicle that can be solved offline based on a nonlinear realistic model of its dynamics. Interestingly, a subdivision of the cooperative merging maneuver into different phases is adopted also in [4], addressing merging of multiple vehicles from a ramp into the freeway. However, in [4] the merging point is fixed and determined by the road configuration, whereas in [9] it is optimized during the coordination phase.

In this work, we consider the same set-up of [9] and focus on the coordination phase. Indeed, despite this phase

All authors are with the Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, 20133 Milano, Italy. E-mail: alessandro.falsone@polimi.it, beatrice.melani@mail.polimi.it, maria.prandini@polimi.it

makes use of a simple double-integrator model of the vehicle dynamics, the computational burden scales with the number of vehicles in the platoon since the optimal merging position has to be identified. We thus propose to pre-compute offline an explicit solution, so as to minimize the onboard computational load. Unfortunately, multi-parametric programming cannot be directly applied to the optimization problem in [9] since the number of vehicles in the platoon is not known in advance. A first contribution of this paper is thus to consider an infinitely long platoon and devise a method to compute the optimal coordination strategy – which is now independent of the number of vehicles –, as an explicit function of the velocity of the merging vehicle and of its relative position along the lane with respect to the adjacent pair of vehicles in the platoon. Since the optimal merging position may change depending on the relative position and speed of the merging vehicle, a second contribution is the introduction of a classifier to establish a-priori the best merging position given the computed explicit maps corresponding to different merging positions. This allows to reduce the onboard memory footprint of the coordination phase, as only portions of the explicit maps corresponding to different merging positions has to be stored, and provides also computational savings, as there is no need to compute and compare (online) the cost functions of the different merging positions to decide where to merge.

II. Formulation of the coordination problem

We consider a platoon of m vehicles driving at some constant speed v_{des} from left to right on the same lane of a straight one-way two-lanes road. The ego vehicle travelling along the same direction but in the adjacent lane needs to change lane and enter the the platoon. Vehicles in the platoon are numbered from 1 to m left to right, and the ego vehicle is assigned number 0 (see Figure 1).

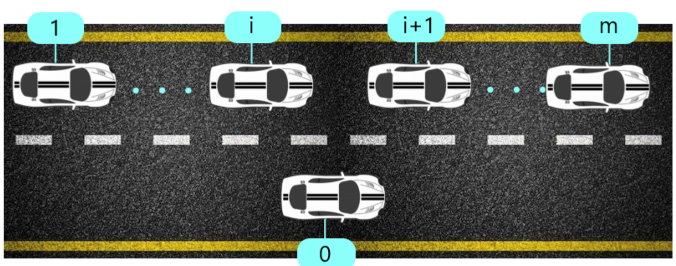


Fig. 1: Reference setting.

Vehicles in the platoon keep an inter-vehicle *recommended* safety distance, which is assumed to be proportional to their velocity v_{des} through a parameter t_{gap} modeling some intervention time in case of system failure.

Since during the coordination phase all vehicles, including the merging one, keep traveling on a straight path along the lane direction, their dynamics can be described by a linear double-integrator model:

$$\dot{x}^i = v^i, \quad \dot{v}^i = a^i, \tag{1}$$

where x^i is the center of mass position of vehicle i with respect to a Cartesian absolute reference frame in which the x axis is taken along the lane. The velocity v^i and the acceleration a^i are both aligned with x.

The goal of the coordination phase is to establish the most convenient merging position and the optimal trajectories of all vehicles that allow the ego vehicle to enter safely when performing the lane change. For the whole duration of the coordination phase, a *minimum* safety distance $v^i\, t_{\mathrm{gap}}^{\min}$, with $t_{\mathrm{gap}}^{\min} < t_{\mathrm{gap}}$ modeling a minimal intervention time, has to be guaranteed between each pair of vehicles $(i, i+1)$ in the platoon, while speed limits $v^i \in [v_{\min}, v_{\max}]$ and also constraints on the acceleration $a^i \in [a_{\min}, a_{\max}]$ are enforced for all vehicles including the ego one, so as to avoid passengers to experience discomfort because of excessive acceleration/deceleration.

Let $j = 0, \ldots, m$ identify the merging scenario, with j set equal to 0 if the ego vehicle enters the platoon at its tail, to m if at its head, and $j = i$, $i = 1, \ldots, m-1$, if between vehicles i and $i+1$. The coordination phase ends when all vehicles reach the v_{des} velocity and vehicle 0 satisfies the merging conditions:

- the distance between vehicle 0 and vehicle $i = j$ is no-smaller than $v^j\, t_{\mathrm{gap}}^{\min}$, if $j > 0$;
- the distance between vehicle 0 and vehicle $i = j+1$ is no-smaller than $v^0\, t_{\mathrm{gap}}^{\min}$, if $j < m$.

Performance is evaluated in terms of time for the coordination phase to be completed. In the sequel, we shall refer to the required final velocity and the merging conditions as *target conditions* for brevity. Note that when the merging maneuver by the ego vehicle is over, the recommended safety distance in the platoon can be recovered via longitudinal control, [10].

Motivated by the adoption of a zero-order-hold converter for the control law implementation, we introduce a discretization with a sampling interval of dt seconds in which the decision variables (the accelerations) are kept constant, i.e.,

$$a^i(t) = a^i_k,\ t \in [kdt, (k+1)dt), \quad k \in \mathbb{N}. \tag{2}$$

The sampled version of the dynamics (1) is then given by

$$x^i_{k+1} = x^i_k + v^i_k dt + \frac{1}{2} a^i_k dt^2, \quad v^i_{k+1} = v^i_k + a^i_k dt \tag{3}$$

where $v^i_k = v^i(kdt)$ and $x^i_k = x^i(kdt)$ are the values of $v^i(t)$ and $x^i(t)$ at time kdt, $k \in \mathbb{N}$.

The minimum time coordination problem enforcing the above mentioned constraints for a given merging scenario j can thus be formulated as the following optimal control problem over the finite horizon $[0, N]$:

$$\begin{aligned}
\min_{\boldsymbol{a},\boldsymbol{h}} \quad & \sum_{k=0}^{N} \left[k\, dt\, h_k + \epsilon_h h_k^2\right] + \epsilon_a \sum_{k=0}^{N-1} \sum_{i=0}^{m} {a^i_k}^2 && \qquad (4)\\
\text{s.t.:} \quad & \text{eq. (3)}, && \forall i,\ k < N\\
& v_{\min} \le v^i_k \le v_{\max}, && \forall i,\ \forall k\\
& a_{\min} \le a^i_k \le a_{\max}, && \forall i,\ k < N\\
& v^i_k\, t_{\mathrm{gap}}^{\min} \le x^{i+1}_k - x^i_k, && 0 < i < m,\ \forall k\\
& v^j_k\, t_{\mathrm{gap}}^{\min} \le x^0_k - x^j_k + h_k, && \forall k,\ \text{if } j > 0\\
& v^0_k\, t_{\mathrm{gap}}^{\min} \le x^{j+1}_k - x^0_k + h_k, && \forall k,\ \text{if } j < m\\
& |v^i_k - v_{\mathrm{des}}| \le h_k, && \forall i,\ \forall k
\end{aligned}$$

where $\boldsymbol{a} = [a_0^0 \cdots a_0^m \cdots a_{N-1}^0 \cdots a_{N-1}^m]^\top$ and $\boldsymbol{h} = [h_0 \cdots h_N]^\top$ are the decision vectors. Vector $\boldsymbol{a}$ is the collection of the accelerations of all vehicles, whereas vector $\boldsymbol{h}$ is the collection of the auxiliary (non-negative) decision variables h_k introduced to measure at each time instant kdt, $k = 1, \dots, N$, the (maximum) violation of the last three constraints in (4), which are associated to the target conditions. The smaller h_k, the closer the vehicles are to the target conditions. The term $k\,dt\,h_k$ in the cost function of (4) increasingly penalizes h_k as time progresses, thus encouraging the attainment of the target conditions in minimal time.

Remark 1: Note that $v_0^i = v_{\text{des}}$ and $x_0^{i+1} - x_0^i = v_{\text{des}} t_{\text{gap}}$, $i = 1, \dots, m$, are the initial velocity and relative position of the vehicles in the platoon, so that – for each given platoon length m – the optimization problem (4) is parametric in the initial velocity v_0^0 and position x_0^0 of vehicle 0 only.

The optimal solution $(\boldsymbol{a}_j^\star, \boldsymbol{h}_j^\star)$ of (4) clearly depends on j. Vehicle 0 should then solve the quadratic program (4) for all values $j = 0, \dots, m$ so as to find the best merging point $j^\star$. Note that trajectories corresponding to $\boldsymbol{a}_{j^\star}^\star$ can then be computed via (3) and tracked by a mid-level controller onboard of the automated vehicles, so as to cope with possible uncertainty affecting their dynamics and initial conditions.

When m is large, the exploration of all values for $j = 0, \dots, m$ can hamper the online implementation of the coordination phase also because the complexity of (4) scales with m too. In [9], this issue is pointed out and distributed schemes (e.g., [11], [12]) for the iterative solution of (4) are suggested, which, however, may require excessive time to converge. We next propose an alternative solution, which allows to move the computation effort offline by pre-computing the optimal merging point $j^\star$ and acceleration profiles $\boldsymbol{a}_{j^\star}^\star$ as a function of the initial velocity and relative position of vehicle 0 with respect to the platoon.

III. Explicit coordination strategy

In this section, we propose an approach to find an explicit solution for the coordination phase that is independent of the actual platoon length m and parametric in the initial velocity and position of the merging vehicle (cf. Remark 1), so that it can be easily stored on-board for direct online application. To determine such a solution, we shall consider an infinite platoon of vehicles and identify the position of the ego vehicle along the x axis in term of its relative position with respect to the two vehicles in the platoon that are closer to it along the x lane direction. The resulting solution will be valid also in the finite-platoon case, since it meets the target merging conditions and satisfies all safety and actuation constraints. This comes at the expense of performance degradation that is expected to be limited and confined to initial positions of the ego vehicle that are close to either the head or the tail of the actual (finite) platoon.

As for the offline computation of the proposed solution, the quadratic optimization problem (4) should be in principle solved with an infinite number m of vehicles, which is not feasible in practice. The idea is then to identify the smallest (even, for convenience) number m° of vehicles such that the optimal acceleration of the first and last vehicles of the platoon is identically zero, for all initial positions x of the merging vehicle between vehicles $\frac{m^\circ}{2}$ and $\frac{m^\circ}{2}+1$ and for all its initial velocities. This indeed entails that the first and last vehicles of the platoon are not affected by the maneuver and neither will be further vehicles added to the tail and/or head. Note that for such a finite value m° to exist, the minimum safety distance must be strictly smaller than the recommended safety since otherwise a sort of domino effect will occur, getting every vehicle involved in the maneuver.

To identify m°, one needs to explore increasing (even) values of m. More precisely, for each tentative value m, the multi-parametric programming problem (4) has to be solved to derive the optimal cost and acceleration profiles as an explicit function of the parameters:

- initial position along the x axis of the ego vehicle ranging between the position of vehicles $\frac{m}{2}$ and $\frac{m}{2}+1$, expressed in terms of the relative position δx_0:
$$\delta x_0^0 = x_0^0 - \frac{x_0^{\frac{m}{2}} + x_0^{\frac{m}{2}+1}}{2} \in \left[-\frac{v_{\text{des}} t_{\text{gap}}}{2}, \frac{v_{\text{des}} t_{\text{gap}}}{2}\right],$$
- initial velocity v_0^0 of the ego vehicle ranging within some interval $[v_{\min}^0, v_{\max}^0] \subseteq [v_{\min}, v_{\max}]$,

for $j = \frac{m}{2} + p$, where p denotes the merging point relative to the pair of vehicles $\frac{m}{2}$ and $\frac{m}{2} + 1$ and takes values in $\{0, \pm 1, \pm 2, ..., \pm \frac{m}{2}\}$. In particular, for each value of p, a piecewise affine map on the initial parameter set

$$\Theta = \left[-\frac{v_{\text{des}} t_{\text{gap}}}{2}, \frac{v_{\text{des}} t_{\text{gap}}}{2}\right] \times [v_{\min}^0, v_{\max}^0]$$

that provides the optimal acceleration profiles for all vehicles along the time horizon $[0, N]$ is obtained.

The maps constructed for different values of p must be combined taking the minimum of the associated costs in order to build the optimal coordinated maneuver map $\mathcal{A}_m : \ \Theta \to \mathbb{R}^{(m+1)N}$ and check if the resulting accelerations of the first and last vehicles of the platoon are identically zero. In practice, one can avoid to build the whole $\mathcal{A}_m$ and just refer to a fine grid of points within the parameter set Θ, given that the costs are piecewise quadratic continuous as a function of the parameters, [13]. When m° is identified, then, $\mathcal{A}^\circ = \mathcal{A}_{m^\circ}$ can be efficiently built by gridding Θ and labeling each grid point with the corresponding optimal value of p. The so-obtained labeled data can be used to train a multiclass Support Vector Machine (SVM) classifier, [14], and determine suitable separators for partitioning Θ. Each element of the partition is then provided by the map associated with the corresponding optimal merging point.

Consider now a platoon of length m with the ego vehicle not necessarily positioned in the middle. We next explain how to retrieve the accelerations of the ego and platoon vehicles from the map $\mathcal{A}^\circ$ computed based on the reference platoon of length m° with the ego vehicle initialized between the pair $(\frac{m^\circ}{2}, \frac{m^\circ}{2} + 1)$ of platoon vehicles.

Define $\bar{\imath} \in \mathbb{Z}$ such that

$$x_0^1 + (\bar{\imath} - 1) t_{\text{gap}} v_{\text{des}} \leq x_0^0 < x_0^1 + \bar{\imath}\, t_{\text{gap}} v_{\text{des}},$$

which, when $1 \leq \bar{\imath} \leq m-1$, means that the ego vehicle is initialized between the pair of vehicles $(\bar{\imath}, \bar{\imath}+1)$, corresponding to pair $(\frac{m^\circ}{2}, \frac{m^\circ}{2}+1)$ in the reference platoon. Then, vehicle i, $i = 1, \ldots, m$, in the actual platoon has a corresponding vehicle $h = i - (\bar{\imath} - \frac{m^\circ}{2})$ in the reference platoon if $1 \leq i - (\bar{\imath} - \frac{m^\circ}{2}) \leq m^\circ$.

Let $[\bar{a}_0^0 \cdots \bar{a}_0^{m^\circ} \cdots \bar{a}_{N-1}^0 \cdots \bar{a}_{N-1}^{m^\circ}]^\top = \mathcal{A}^\circ(\delta x_0^0, v_0^0)$, with $\delta x_0^0 = x_0^1 + \frac{2\bar{\imath}-1}{2} t_{\mathrm{gap}} v_{\mathrm{des}}$, be the optimal acceleration of all vehicles in the reference platoon. Then, for $k = 0, \ldots, N-1$, the acceleration profiles of vehicles $i = 1, 2, \ldots, m$ in the actual platoon will be given by

$$a_k^i = \begin{cases} \bar{a}_k^{i-\bar{\imath}+\frac{m^\circ}{2}}, & \text{if } 1 \leq i - \bar{\imath} + \frac{m^\circ}{2} \leq m^\circ \\ 0, & \text{otherwise,} \end{cases} \tag{5}$$

while that of the ego vehicle will be $a_k^0 = \bar{a}_k^0$.

In the next section, we compare the proposed explicit coordination strategy with the online optimization approach in [9] on a numerical example.

IV. Comparative performance analysis

We consider the case when vehicles in the platoon are traveling at $v_{\mathrm{des}} = 70$ km/h[1], and the speed limits are $v_{\min} = 0$ km/h and $v_{\max} = 90$ km/h. We set the safety parameters as $t_{\mathrm{gap}} = 1.5$ s and $t_{\mathrm{gap}}^{\min} = 1$ s. An absolute reference frame centered on the initial position x_0^1 of the tail vehicle is considered when formulating problem (4). Positive values of x are in the direction of motion. Initial positions and velocities of the platoon vehicles are set in the absolute reference frame as $x_0^i = (i-1) t_{\mathrm{gap}} v_{\mathrm{des}}$ and $v_0^i = v_{\mathrm{des}}$, respectively, for all $i = 1, \ldots, m$. The sampling time is $dt = 1$ s and the control horizon is $T = 10$ s, thus obtaining a finite horizon of $N = 10$ time slots. The weights in the cost function in (4) are set as $\epsilon_a = 0.1$ and $\epsilon_h = 0.001$. Comfort constraints are $a_{\min} = -3$ m/s^2 and $a_{\max} = 2$ m/s^2. The parameters δx_0^0 and v_0^0 identifying the initial conditions of vehicle 0 are ranging in Θ with $v_{\min}^0 = 30$ km/h (a reasonable value of speed for a vehicle in an on-ramp) and $v_{\max}^0 = v_{\max} = 90$ km/h. Strategies are implemented in MATLAB R2019b on a laptop with an Intel Core i5-6267U CPU and 4 GB of RAM.

Computation of the explicit coordination strategy

The MPT3 toolbox [15] is used for solving the multi-parametric optimization problem (4) incrementally in m according to the procedure described in Section II to finally obtain the value $m^\circ = 14$ for the number of vehicles approximating an infinite platoon. Figure 2 reports the maps that have to be combined for determining the optimal coordinated maneuver map $\mathcal{A}^\circ = \mathcal{A}_{m^\circ}$. They correspond to $p = -1, 0, 1$, meaning that the ego vehicle should enter either between the pair $\frac{m^\circ}{2}$ and $\frac{m^\circ}{2}+1$ of platoon vehicles ($p = 0$) or between the preceding ($p = -1$) and consequent ($p = 1$) pairs, the best merging point depending on its initial conditions.

To obtain $\mathcal{A}^\circ$, the three maps are sampled with a uniform grid (0.5 m along the horizontal axis and 1.8 km/h along the vertical axis). A label is associated with every point of the grid indicating the value of the merging point p with the lowest cost. This results in the grid in Figure 3 in which three different regions associated to the three different labels are easily distinguishable.

The interpretation of the combined map on the infinite platoon is intuitive. Let $(\bar{\imath}, \bar{\imath}+1)$ be the pair of platoon vehicles closer to the ego vehicle at the beginning of the maneuver. When v_0^0 is lower than $v_{\mathrm{des}} = 70$ km/h and x_0^0 is close enough to $x_0^{\bar{\imath}}$, it is convenient to merge the platoon behind vehicle $\bar{\imath}$ (green zone, corresponding to $p = -1$). Similarly, when v_0^0 is higher than $v_{\mathrm{des}} = 70$ km/h and x_0^0 is close enough to $x_0^{\bar{\imath}+1}$ it is better for the ego vehicle to merge the platoon in front of vehicle $\bar{\imath}+1$ (red region, corresponding to $p = 1$). In all the in-between cases, it is convenient for the merging vehicle to enter the platoon between $\bar{\imath}$ and $\bar{\imath}+1$ (yellow zone, corresponding to $p = 0$). Moreover, the regions are not symmetric because the range of velocities is itself not symmetric with respect to the value of v_{des} and the constraints on the acceleration are also not symmetric. Note that in the extreme case in which the ego vehicle is initialized beside one vehicle in the platoon, the choice of the closest pair of platooning vehicles is not unique and, as such, the optimal coordinating strategy corresponding to the extreme positions in the combined map is identical.

The points of the map in Figure 3 are used to train two binary SVM classifier, in order to find the separators between classes $p = -1$ and $p = 0$ and classes $p = 0$ and $p = 1$. The Classification Learner app for Matlab is used to this purpose, using all the observations with labels 0 and -1 as entries for the training of the first classifier and all the ones with labels 0 and 1 for the second one. In both cases, among the tested linear, quadratic, cubic, and Gaussian SVN classifiers, the cubic one provides the best accuracy on the training data. The obtained cubic SVN classifiers (black lines in Figure 3) are then tested on a grid of points with the same structure as the previous one, but a bit shifted to have different data. The first classifier wrongly labels 3 points over 1863, while the second one 2 over 1584, which are both reasonably low percentages (0.16% and 0.126%, respectively), also taking into account the limited impact of a wrong labeling on the frontier, since the frontier is defined taking the minimum of two (piecewise quadratic) continuous cost functions, [13].

Explicit solution versus the online optimal one

As anticipated in Section III, the solution found using the parametric approach is independent of the number of vehicles in the platoon, but when applied to the finite platoon a degradation of performance may occur.

We assess the degradation amount by considering platoons of $m \in \{6, 8, 10, 12, 14, 16\}$ vehicles with the initial velocity and position of the ego vehicle set to $v_0^0 = 40$ km/h and $x_0^0(s) = x_0^1 + \frac{s}{4} t_{\mathrm{gap}} v_{\mathrm{des}}$, $s \in S_m = -8, -7, \ldots, 4(m+1)$, i.e., with x_0^0 every quarter of $t_{\mathrm{gap}} v_{\mathrm{des}}$ between $x_0^1 - 2 t_{\mathrm{gap}} v_{\mathrm{des}}$ (far before the platoon tail) and $x_0^m + 2 t_{\mathrm{gap}} v_{\mathrm{des}}$ (far after the platoon head).

In the online case, the cost is computed solving the optimization problem (4) for the given m and initial state

[1] All velocity values are reported in km/h for readability, but we used m/s for the implementation.

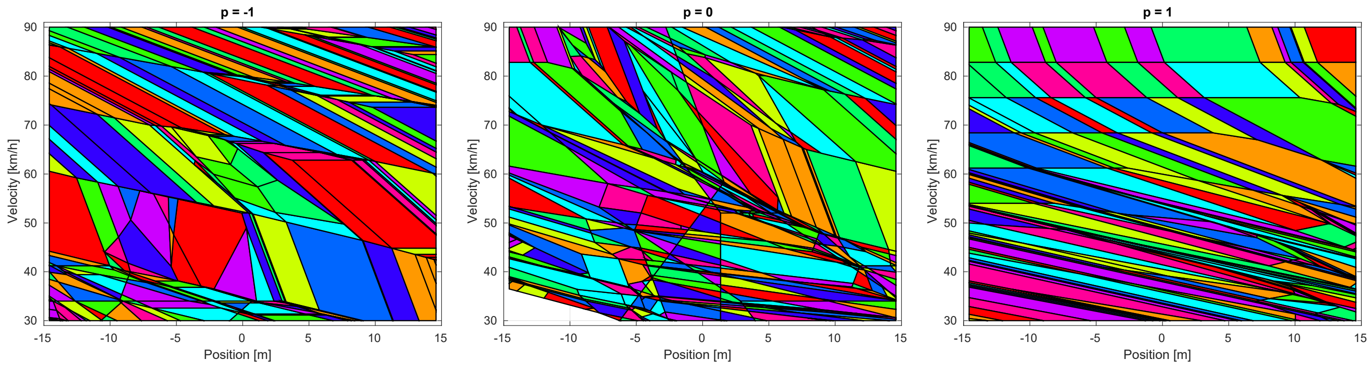


Fig. 2: Polyhedral partitions associated to the optimal coordinated maneuver maps for the merging scenarios with $p = -1$, $p = 0$, $p = 1$, from left to right, obtained via the MPT3 toolbox. Colors are used to distinguish different polyhedra.

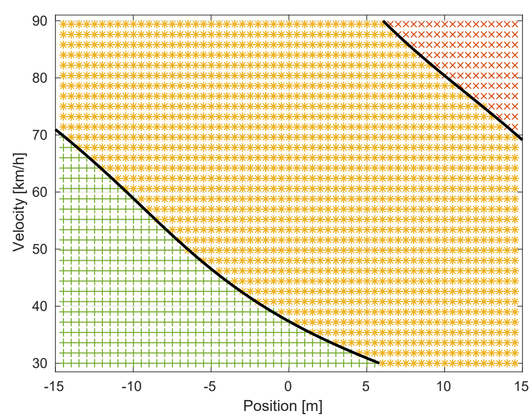


Fig. 3: Grid map with labels $p = -1$ (green), $p = 0$ (yellow), and $p = 1$ (red) and SVM separators (black lines)

.

$(x_0^0(s), v_0^0)$ of the ego vehicle, for each $s \in S_m$, using YALMIP ([16]) with CPLEX 12.10 as solver, as in [9]. Instead, in the explicit solution case, we first evaluate $\mathcal{A}^\circ$ as discussed in Section III, for all $s \in S_m$, which returns the corresponding acceleration profile $a_k^i(s)$, $i = 0, 1, \ldots, m$, from (5). Then, for all m and $s \in S_m$, we compute the cost as the minimum of (4), optimizing over $\boldsymbol{h}$ only, while setting the accelerations equal to $a_k^i(s)$.

By comparing the obtained cost with that of the online optimization case, it emerges that the number of cars in the platoon does not affect the results. In fact, in all instances where the merging vehicles is initialized within the limits of the platoon ($x_0^1 \leq x_0^0 \leq x_0^m$), the cost is almost the same, with a maximum increment of 0.315%. What affects the most the degradation of the cost figure is the initialization point: when the merging vehicle is initialized either at the tail or the head of the platoon, the cost figure obtained using the pre-computed acceleration profiles from the maps in Figure 2 is larger than the one computed online. Nevertheless, the degradation is at most 4.52% for all instances in which at least one vehicle of the platoon is involved in the maneuver. The increment can go up to 54.31% in those cases in which the vehicles in the platoon are not affected by the merging of vehicle 0. However, those are the instances in which applying the acceleration profiles from the maps is no longer meaningful, because platooning vehicles are not affected by the ego vehicle lane change maneuver and the optimization problem in (4) no longer reflects the actual scenario at hand.

Figure 4 shows the trajectories and velocity profiles of a platoon with $m = 10$ cars, when vehicle 0 is initialized in $x_0^0 = x_0^7 + 0.5 t_{\text{gap}} v_{\text{des}}$, the middle point between the positions of vehicle $i = 7$ and $i = 8$, with merging position $p = 0$, meaning that vehicle 0 joins the platoon between vehicles $i = 7$ and $i = 8$. The reported plots for the offline solution are identical to those of the solution computed online and, hence, the cost is the same. As for the computing times, retrieving the optimal acceleration profiles from the pre-computed map, including the time to apply the binary SVN classifier twice, requires about 0.25 seconds, while computing them online requires about 1.98 seconds, which is comparable to the time needed to complete the coordination maneuver (7 seconds).

Figure 5 refers to the case of a platoon of $m = 8$ cars with vehicle 0 initialized in $x_0^0 = x_0^m + 1.25 t_{\text{gap}} v_{\text{des}}$. Given the fact that the initialization is far after the platoon head, the merging occurs ahead of vehicle $i = 8$. The time to determine the optimal acceleration profiles is 0.14 seconds

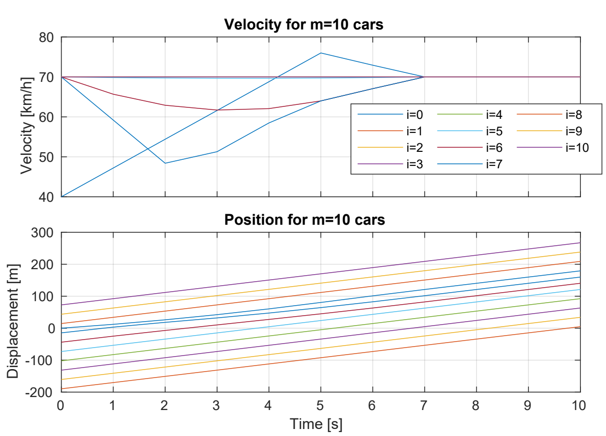
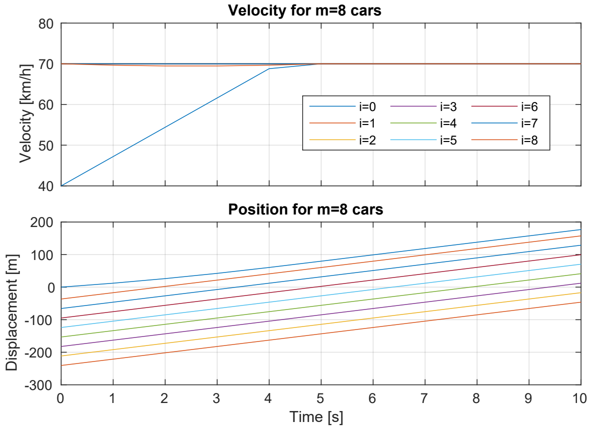
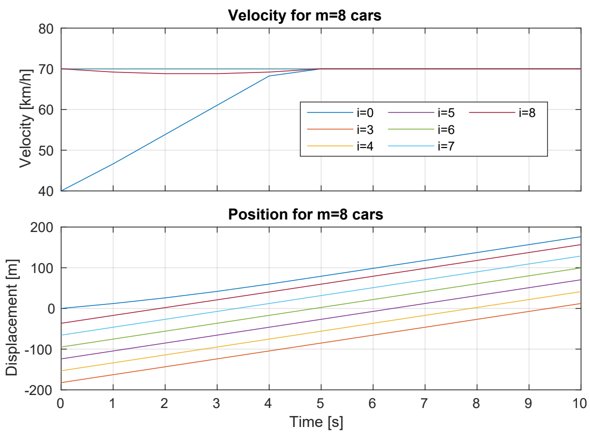


Fig. 4: Explicit solution computed offline based on an infinite platoon ($m = 10$, $x_0^0 = x_0^7 + 0.5 t_{\text{gap}} v_{\text{des}}$).



(a) Optimal solution computed online.



(b) Explicit solution computed offline based on an infinite platoon.

Fig. 5: Comparison of position and velocity profiles ($m = 8$ and $x_0^0 = x_0^{m+1} + 0.25 t_{\mathrm{gap}} v_{\mathrm{des}}$). Vehicles 1 and 2 do not appear in plot (b) since they do not take part to the maneuver (their acceleration is set to zero according to (5)).

(explicit solution) versus 1.6 seconds (online optimization) with a time to complete the coordination maneuver of about 5 seconds in both cases. By looking at the velocity profile of vehicle $i = 8$ in the online and offline solutions in Figure 5, we can notice that they are not the same: in fact, vehicle $i = 8$ in the offline case decelerates more with respect to the same vehicle in the online solution. At the same time, the velocity profile of vehicle 0 is steeper in the online case. Therefore the solutions, despite being similar, are not identical, consistently with the fact that the cost of the offline case is 4.5% higher than the online one. This is due to the fact that, while for the online optimization case it is possible to calculate the solution removing the constraint prescribing a minimum distance between the merging vehicle and vehicle $j + 1$ in (4), in the offline case the same cannot be done because the solution refers to an infinite platoon. For this reason, in the offline case, vehicle 0 accelerates less to maintain a minimum distance from a non-existing vehicle ahead of vehicle 8, and, consequently, vehicle 8 needs to decelerate more to keep the minimum distance from vehicle 0 itself.

V. Conclusion

We propose an explicit solution to the coordination phase of a lane change maneuver where a vehicle has to enter a platoon of vehicles traveling in the adjacent lane at a constant speed. This solution can be integrated in the two-phase merging scheme in [9] to provide an offline solution to the overall automated lane change problem, which is valid irrespective of the number of vehicles in the platoon.

Technological aspects related to sensors and communication systems need to be addressed before the actual implementation of the proposed solution on real vehicles, [3]. Also, a closed loop explicit model predictive control solution could be devised if more parameters were added, namely the position and velocity of all the platoon vehicles, or, at least position and velocity of those involved in the maneuver. This, however, requires further investigation.

References

[1] C.-Y. Chan, “Advancements, prospects, and impacts of automated driving systems,” *Int. Journal of Transportation Science and Technology*, vol. 6, no. 3, pp. 208–216, 2017.

[2] G. Fitch, S. Lee, S. Klauer, J. Hankey, J. Sudweeks, and T. Dingus, “Analysis of lane-change crashes and near-crashes,” *US Department of Transportation, National Highway Traffic Safety Administration*, 2009.

[3] D. Bevly, X. Cao, M. Gordon, G. Ozbilgin, D. Kari, B. Nelson, J. Woodruff, M. Barth, C. Murray, A. Kurt *et al.*, “Lane change and merge maneuvers for connected and automated vehicles: A survey,” *IEEE Trans. on Intelligent Vehicles*, vol. 1, no. 1, pp. 105–120, 2016.

[4] Y. Xie, H. Zhang, N. H. Gartner, and T. Arsava, “Collaborative merging strategy for freeway ramp operations in a connected and autonomous vehicles environment,” *Journal of Intelligent Transportation Systems*, vol. 21, no. 2, pp. 136–147, 2017.

[5] Y. Luo, G. Yang, M. Xu, Z. Qin, and K. Li, “Cooperative lane-change maneuver for multiple automated vehicles on a highway,” *Automotive Innovation*, vol. 2, no. 3, pp. 157–168, 2019.

[6] B. Li, Y. Zhang, Y. Ge, Z. Shao, and P. Li, “Optimal control-based online motion planning for cooperative lane changes of connected and automated vehicles,” in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2017, pp. 3689–3694.

[7] J. Nilsson, M. Brännström, E. Coelingh, and J. Fredriksson, “Longitudinal and lateral control for automated lane change maneuvers,” in *2015 American Control Conference*, 2015, pp. 1399–1404.

[8] D. Yang, S. Zheng, C. Wen, P. J. Jin, and B. Ran, “A dynamic lane-changing trajectory planning model for automated vehicles,” *Transp. Research Part C: Emerging Technologies*, vol. 95, pp. 228–247, 2018.

[9] A. Falsone, B. Sakcak, and M. Prandini, “Coordinated lane change in autonomous driving: a computationally aware solution,” *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 15 211–15 216, 2020.

[10] V. Lesch, M. Breitbach, M. Segata, C. Becker, S. Kounev, and C. Krupitzer, “An overview on approaches for coordination of platoons,” *IEEE Trans. on Intelligent Transp. Systems*, pp. 1–17, 2021.

[11] A. Falsone, K. Margellos, S. Garatti, and M. Prandini, “Dual decomposition for multi-agent distributed optimization with coupling constraints,” *Automatica*, vol. 84, pp. 149–158, 2017.

[12] A. Falsone, I. Notarnicola, G. Notarstefano, and M. Prandini, “Tracking-ADMM for distributed constraint-coupled optimization,” *Automatica*, vol. 117, 2020.

[13] V. Dua, N. A. Bozinis, and E. N. Pistikopoulos, “A multiparametric programming approach for mixed-integer quadratic engineering problems,” *Computers & Chemical Engineering*, vol. 26, pp. 715–733, 2002.

[14] C.-W. Hsu and C.-J. Lin, “A comparison of methods for multiclass support vector machines,” *IEEE Trans. on Neural Networks*, vol. 13, no. 2, pp. 415–425, 2002.

[15] M. Herceg, M. Kvasnica, C. Jones, and M. Morari, “Multi-Parametric Toolbox 3.0,” in *European Control Conf.*, 2013, pp. 502–510.

[16] J. Lofberg, “YALMIP: a toolbox for modeling and optimization in MATLAB,” in *IEEE Int. Conf. on Robotics and Automation*, 2004, pp. 284–289.