2020 59th IEEE Conference on Decision and Control (CDC)
Jeju Island, Republic of Korea, December 14-18, 2020

Linear Differential Games for Cooperative Behavior Planning of Autonomous Vehicles Using Mixed-Integer Programming

Tobias Kessler[1]*, Klemens Esterle[1]* and Alois Knoll[2]

***Abstract*— Cooperatively planning for multiple agents has been proposed as a promising method for strategic and motion planning for automated vehicles. By taking into account the intent of every agent, the ego agent can incorporate future interactions with human-driven vehicles into its planning. The problem is often formulated as a multi-agent game and solved using iterative algorithms operating on a discretized action or state space. Even if converging to a Nash equilibrium, the result will often be only sub-optimal. In this paper, we define a linear differential game for a set of interacting agents and solve it to optimality using mixed-integer programming. A disjunctive formulation of the orientation allows us to formulate linear constraints to prevent agent-to-agent collision while preserving the non-holonomic motion properties of the vehicle model. Soft constraints account for prediction errors. We then define a joint cost function, where a cooperation factor can adapt between altruistic, cooperative, and egoistic behavior. We study the influence of the cooperation factor to solve scenarios, where interaction between the agents is necessary to solve them successfully. The approach is then evaluated in a racing scenario, where we show the applicability of the formulation in a closed-loop receding horizon replanning fashion. By accounting for inaccuracies in the cooperative assumption and the actual behavior, we can indeed successfully plan an optimal control strategy interacting closely with other agents.**

I. Introduction

When sharing the road with human drivers, autonomous vehicles will have to cooperate with other vehicles to fit safely into nowadays traffic scenarios while still asserting their own goals. In dense traffic, where space is often limited, such as highway overtaking or merging, the reactions of others must be anticipated, and a maneuver-based prediction will perform poorly. Instead, a planner must model the uncertain interaction with the other traffic participants by planning a joint action for the ego vehicle and the surrounding vehicles. Fig. 1 depicts an exemplary scenario.

A variety of concepts have been proposed to cope with these challenges, with game-theoretic approaches being capable of modeling the interaction between multiple agents elegantly [1]. However, they often require a discretization of the action or the state space, yielding an approximation of the optimal solution, which are usually work well for a limited set of scenarios. Furthermore, these algorithms often rely on a random sampling of the solution space or lack guarantees of convergence and thus pose open questions towards certification of such systems. Optimal control theory, on the other hand, provides deterministic solution algorithms converging to an optimum and has been successfully applied to safety-critical systems.

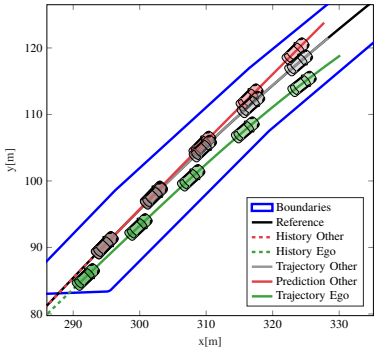


Fig. 1: Example of a multi-agent planning scenario. The green ego agent wants to overtake the red agent but is unaware of its exact future motion (gray). Both agents intend to track the same reference line and must stay inside the road boundaries (blue). Dark-to-light colors depict progressing time.

In this work, we propose an approach how an autonomous agent can safely interact with other agents while still achieving its own goals. For this, we formulate the multi-agent planning problem as a differential game and solve it using mixed-integer quadratic programming (MIQP) with an off-the-shelf solver. Specifically, we contribute a linear differential game formulation featuring

- a set of linear constraints as inter-agent collision check,
- a leveraged joint cost function for collaborative planning, and
- a methodology to handle inaccurate models of other agents using soft constraints.

We first examine the impact of cooperation in a symmetric negotiation example. We then demonstrate the contributions in a competitive race track example as this scenario poses high demands on the accurate modeling of vehicle kinematics, handling of vehicle collisions, and interactive behavior.

II. Related Work

We follow the distinction of Ulbrich *et al.* [2] dividing cooperative driving into explicit inter-vehicle communication and cooperation in the form of collaboration. In Table I we compare some approaches with comparable problem formulations or similar solution methods. For a more comprehensive overview we refer to Schwarting *et al.* [1].

Game-theoretic formulations can be employed to model collaboration. A decision tree often realizes the game-theoretic setting. Kessler and Knoll [3] use motion primitives

*These authors contributed equally to this work.

[1]Tobias Kessler and Klemens Esterle are with fortiss GmbH, Research Institute of the Free State of Bavaria, Munich, Germany, surname@fortiss.org

[2]Alois Knoll is with Robotics, Artificial Intelligence and Real-time Systems, Technische Universität München, Munich, Germany

978-1-7281-7447-1/20/$31.00 ©2020 IEEE

TABLE I: Comparison of different solution approaches for multi-agent planning using the abbreviations mixed-integer quadratic programming (MIQP), mixed-integer linear programming (MILP), Monte Carlo tree search (MCTS), dynamic programming (DP).

Source	Interactiveness	Action Space	State Space	Solution Method	Globalization Strategy
Kessler and Knoll [3]	Global cooperative costs	discrete	cont.	MILP	W.r.t. sampled motion tree
Lenz *et al.* [4]	Multi-agent dynamic game, non-zero-sum	discrete	cont.	MCTS	None
Bahram *et al.* [5]	Two-player dynamic game	discrete	cont.	Alpha-beta pruning	None
Liniger and Lygeros [6]	Multi-agent dynamic game, non-zero-sum	cont.	cont.	DP	Solving Nash equilibrium
Schwarting *et al.* [7]	Multi-agent dynamic game, non-zero-sum	cont.	cont.	Iterative DP	Solving iterative Nash equilibrium
Fabiani and Grammatico [8]	Potential game	mixed	mixed	MIQP	Solving iterative Nash equilibrium
Eilbrecht and Stursberg [9]	Global cooperative costs, shared plans	cont.	cont.	MIQP	Iterative conflict resolution
Manzinger and Althoff [10]	Centralized planning, shared plans	cont.	discrete	Reachability	Auction-based conflict resolution
Frese and Beyerer [11]	Global cooperative costs, centralized	cont.	cont.	MILP	Inherently
Esterle *et al.* [12]	None	cont.	cont.	MIQP	Inherently

to generate a motion tree and use mixed-integer linear programming (MILP) to solve for the optimal orchestration. While it is applicable in arbitrary environments, the action space discretization can exclude the optimal solution, yielding difficulties when applying to dense traffic scenarios. They account for unknown cost functions of other agents by updating the costs based on observations. Monte Carlo tree search can be used to plan collaborative behavior, effectively solving a multi-agent, non-zero-sum dynamic game [4]. A cooperation factor serves as a tuning parameter in the ego agents' cost function. The formulation is highly flexible and can incorporate any transition function for modeling the environment. However, the discretized action space yields similar problems to [3]. An extensive-form game is formulated in [5], where the other traffic participants are modeled as part of the environment. While the framework is highly flexible and has proven to work in a real car under real-time requirements, the approach does not ensure convergence to an optimal solution. A two-player dynamic, non-zero-sum game is formulated as a bimatrix game in [6], which allows for an efficient calculation of the Nash equilibrium. Schwarting *et al.* [7] use iterative dynamic programming to solve a multi-agent dynamic, non-zero-sum game. They explicitly model partial observability of the intention of others. The proposed believe-space variant of the iterative Linear Quadratic Gaussian (iLQG) algorithm can be executed in real-time. Exact costs and dynamics of other agents are assumed to be known, and the algorithm converges to a potentially sub-optimal Nash equilibrium. Multi-vehicle driving as a potential game is formulated in [8] and solved using MIQP. The potential function allows the authors to compute a ϵ-mixed-integer Nash equilibrium. However, the discrete lateral action and state space complicate applying this approach in reality.

For explicit communication between vehicles, the planning problem effectively simplifies, as accounting for the unknown intentions of other agents becomes irrelevant. Eilbrecht and Stursberg [9] formulate a two-layered approach of iterative conflict resolution using a cooperative cost function. The underlying behavior of each agent is generated through an optimal control problem using MIQP for each agent while ensuring obstacle avoidance to the (known) plan of the other agents. However, their approach is only valid for straight driving on straight roads, and the iterative conflict resolution does not offer any guarantees to converge to a global optimum. Manzinger and Althoff [10] use reachability analysis to compute conflicting space-time cells, which might be occupied by multiple vehicles. An auction algorithm then solves for those conflicts. Their approach requires a discretization of the state space. Various approaches exist to minimize a cooperative cost function based on the premise that the cost functions of the other agents are known. Frese and Beyerer [11] formulate a multi-agent optimal control problem and solve it using MILP. However, their approach is limited to straight driving on straight roads and yields a lot of invalid solutions otherwise. Our previous work [12] solves those issues but was not defined for a multi-agent context.

III. Problem Formulation and Assumptions

We aim to find a control strategy in an environment with multiple agents, subject to the following set of assumptions. Firstly, we only control one agent but indirectly assume perfect observation of the dynamic state of other agents, as well as their goals. Secondly, we assume a perfect perception of the map and our localization within that. And thirdly, that other agents do not behave destructively, for example, do not aim for collisions. The problem then consists of

- multiple agents with continuous actions,
- a scene consisting of all individual states, road geometry, and obstacle information,
- each agent tries to achieve its own goals, without adversarial behavior, and
- perfect observation of states at t_0.

We formulate this as a multi-agent, non-zero-sum, differential game [13] with

- a set of agents $\mathscr{A}$ with continuous actions, and
- a joint, non-zero-sum, cost function.

The strategy is executed in a receding horizon fashion. We discretize the time horizon of one iteration into N steps with a time interval Δt. We denote the discrete time step by k and the interval of N steps by $\mathscr{K}$. The goal is thus to find a sequence of actions for the ego vehicle that minimizes its costs while avoiding collisions with the environment.

IV. Linear Dynamic Game using MIQP

In this section, we formulate the differential game and then solve it using MIQP. The linear constraint formulation yields a linear differential game. Each agent in the optimization problem satisfies the constraints from our previous work [12], which we discussed in Section IV-A. Also, we introduce linear constraints for agent-to-agent collision checking in Section IV-B, and a joint cost function in Section IV-C.

A. Preliminaries

The model in this paper builds upon our previously proposed model. In the following, we will highlight the contributions of our previous work; for a detailed model formulation we refer to Esterle *et al.* [12].

a) Linear Model: We model the vehicle as a third-order point-mass system in a global Cartesian frame with positions $p_x(k)$, $p_y(k)$, velocities $v_y(k)$, $v_y(k)$, and accelerations $a_x(k)$, $a_y(k)$ as states. Jerk in both directions $j_x(k)$ and $j_y(k)$ are the inputs of the model. This model is subject to a set of constraints for collision avoidance and non-holonomy.

Although the vehicle's orientation θ is not part of the state space, we need it for a sufficient collision check in the Cartesian coordinates within the optimization problem. We define *regions* in the (v_x, v_y) plane, allowing us to obtain an approximation of the orientation based on linear inequalities only dependent on v_x and v_y. Note that computing the true orientation $\theta = \text{atan}(v_y/v_x)$ would introduce non-linearity. A binary decision variable ρ defines in which region the vehicle is at each time step. It is set by constraints based on the upper and lower linear bounds of each region. We impose region-dependent limits on acceleration and jerk.

We approximate the collision shape of the vehicle using circles. We over-approximate the front axle collision shape because the true front axle position (f_x,f_y) is not directly known from the linear model. Computing the upper and lower bounds for sine and cosine functions of the orientation leads to upper $\overline{f_x}$, $\overline{f_y}$ and lower bounds $\underline{f_x}$, $\underline{f_y}$ for the x and y position of the front axle. Fig. 2 illustrates the approximation. For obtaining the lower and upper bounds for sine and cosine functions of the true orientation, we fit linear polynomials on v_x and v_y.

b) Modeling the Non-Holonomics: To ensure non-holonomics of a vehicle for a point-mass system, we constrain the curvature, that is highly non-linear and thus cannot be expressed as a linear constraint in MIQP. We thus model the non-holonomics as linear constraints on the acceleration in the y-direction, for which we provide the theory. For that, the curvature is approximated using a linear combination of v_x, v_y, a_x, and is solved for a_y.

c) Environment and Obstacle Collision Avoidance: The polygonal representation of the environment is deflated with the radius of the collision circles. Non-convex environment polygons are split into several convex sub-polygons. We enforce the vehicle to be in at least one of these convex sub-polygons. This way, the vehicle-to-environment-polygon collision check becomes a point-to-polygon check for each circle used to approximate the vehicle.

We ensure the vehicle not to collide with an arbitrary number of static or dynamic convex obstacle polygons, such as obtained from a prediction. However, our previous work does not implement a vehicle-to-vehicle collision check, which we contribute in this work.

B. Multi-Agent Collision Constraints

We approximate the shape of each vehicle to formulate an agent-to-agent collision constraint. In our linear model, the actual orientation of the vehicle is not available, but with the region-based formulation, we can compute a lower- and upper-bounding rectangle for the center of the front axle.

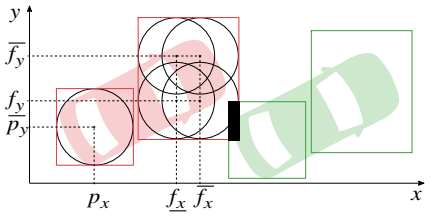


Fig. 2: Approximation of the vehicle shape to formulate the agent-to-agent collision check based on the rectangles of the respective agents. The black area indicates a collision.

In the following, the superscript $\square^i$ refers to the respective variable of the agent i. For collision avoidance, we approximate the vehicle shape by circles with radius R^i, one around the rear axle center, and four for the front axle approximation. To avoid agent-to-agent collisions, we choose to over-approximate these circles with axis-aligned squares again, as sketched in Fig. 2. A better approximation of the circles, for example, with two rectangles, yields more constraints and binary variables, lowering the runtime without providing a huge benefit. We formulate four sets of constraints, the first prevents collisions between the rear parts of two agents, the second prevents collisions between the rear part of the first and the front part of the second agent, the third vice versa, and the fourth prevents collisions between the front parts of both agents for each pair of agents.

The rear part-to-rear part collision constraint of two agents A^i and A^j is based on the following logical formula. We define the sum of both radii $R^{i+j} := R^i + R^j$. A collision occurs at one time step k if and only if

$$\begin{aligned} &p_x^i(k) \geq p_x^j(k) - R^{i+j} \wedge p_x^i(k) \leq p_x^j(k) + R^{i+j} \\ \wedge\ &p_y^i(k) \geq p_y^j(k) - R^{i+j} \wedge p_y^i(k) \leq p_y^j(k) + R^{i+j}. \end{aligned} \tag{1}$$

Intuitively, (1) states that a collision occurs if both the absolute distance in x-direction $|p_x^i(k) - p_x^j(k)|$ and y-direction $|p_y^i(k) - p_y^j(k)|$ is smaller R^{i+j}. Logical negation yields that two agents do not collide at time step k if and only if

$$\begin{aligned} &p_x^i(k) \leq p_x^j(k) - R^{i+j} \vee p_x^i(k) \geq p_x^j(k) + R^{i+j} \\ \vee &p_y^i(k) \leq p_y^j(k) - R^{i+j} \vee p_y^i(k) \geq p_y^j(k) + R^{i+j}. \end{aligned} \tag{2}$$

We formulate this as linear constraints using a set of four decision variables $\alpha_{\square}^{ij}$, one for each inequality and an appropriately chosen big constant M.

$$p_x^i(k) \leq p_x^j(k) - R^{i+j} + M\alpha_1^{ij}(k) \tag{3a}$$
$$p_x^i(k) \geq p_x^j(k) + R^{i+j} - M\alpha_2^{ij}(k) \tag{3b}$$
$$p_y^i(k) \leq p_y^j(k) - R^{i+j} + M\alpha_3^{ij}(k) \tag{3c}$$
$$p_y^i(k) \geq p_y^j(k) + R^{i+j} - M\alpha_4^{ij}(k) \tag{3d}$$
$$3 \geq \sum_{a=1}^{4} \alpha_a^{ij}(k)\ \forall\ k \in \mathcal{K}. \tag{3e}$$

(3e) represents the logical formulas (2) by coupling the four constraints (3a) - (3d) and makes sure no more than three are active, and hence no rear part-to-rear part collision occurs.

As we aim to cope with agents that are not controlled by our algorithm (e.g., human-driven vehicles), the computed motion will not exactly match the reality. This yields prediction errors for the uncontrolled agents, which then can lead to infeasible optimization problems or imminent collisions. To account for these prediction errors, we introduce an additional safety distance for the ego agent to all uncontrolled agents as a *soft constraint*. Inspired by the formulation in [14], we introduce *slack variables* ξ to the agent-to-agent collision constraints. With the desired safety distance of both agents $D(k)$ and a set of slack variables $\xi_x(k), \xi_y(k) \in [0,\ D(k)]$, we modify (3) to

$$p_x^i(k) \le p_x^j(k) - R^{i+j} - D(k) + \xi_x^{ij}(k) + M\alpha_1^{ij}(k) \tag{4a}$$
$$p_x^i(k) \ge p_x^j(k) + R^{i+j} + D(k) - \xi_x^{ij}(k) - M\alpha_2^{ij}(k) \tag{4b}$$
$$p_y^i(k) \le p_y^j(k) - R^{i+j} - D(k) + \xi_y^{ij}(k) + M\alpha_3^{ij}(k) \tag{4c}$$
$$p_y^i(k) \ge p_y^j(k) + R^{i+j} + D(k) - \xi_y^{ij}(k) - M\alpha_4^{ij}(k) \tag{4d}$$
$$3 \ge \sum_{a=1}^{4} \alpha_a^{ij}(k)\ \forall\ k \in \mathcal{K}. \tag{4e}$$

The slack variables will be included in the cost function (see Section IV-C). The optimizer will then seek to keep the slack variables as small as possible. Consequently, in (4), the additional safety distance D will be as high as possible. With this concept, fatal prediction errors (immanent collisions) are mitigated. Note that adding a hard safety margin only leads to more conservative behavior and does not avoid infeasible optimization problems.

To prevent collisions between the rear part of agent A^i and the front part of agent A^j, we again formulate logical constraints that a collision occurs at k if and only if

$$\begin{aligned} p_x^i(k) \ge \underline{f_x}^j(k) - R^{i+j} \wedge p_x^i(k) \le \overline{f_x}^j(k) + R^{i+j} \\ \wedge\ p_y^i(k) \ge \underline{f_y}^j(k) - R^{i+j} \wedge p_y^i(k) \le \overline{f_x}^j(k) + R^{i+j}. \end{aligned} \tag{5}$$

Here we force the point $(p_x^i,\ p_y^i)$ to be outside the front axle approximation rectangle enlarged by the sum of the collision circle radii. The set of constraints is formulated as described for the rear part-to-rear part collision case. Another analog set of constraints prevents collisions between the rear part of A^j and the front part of A^i.

We avoid collisions between the fronts of agents A^i and A^j applying the same strategy but forcing the center point of the front axle approximation rectangle of A^j to retain sufficient distance to the front axle approximation rectangle of A^i. Concretely, we define the sufficient distance as R^{i+j} plus the size of the approximation rectangle of agent A^j. Hence, no front part-to-front part collision occurs at time step k if and only if

$$\begin{aligned} &\frac{1}{2}(\overline{f_x}^j(k) + \underline{f_x}^j(k)) \le \underline{f_x}^i(k) - R^{i+j} - \frac{1}{2}(\overline{f_x}^j(k) - \underline{f_x}^j(k)) \vee \\ &\frac{1}{2}(\overline{f_x}^j(k) + \underline{f_x}^j(k)) \ge \overline{f_x}^i(k) + R^{i+j} + \frac{1}{2}(\overline{f_x}^j(k) - \underline{f_x}^j(k)) \vee \\ &\frac{1}{2}(\overline{f_y}^j(k) + \underline{f_y}^j(k)) \le \underline{f_y}^i(k) - R^{i+j} - \frac{1}{2}(\overline{f_y}^j(k) - \underline{f_y}^j(k)) \vee \\ &\frac{1}{2}(\overline{f_y}^j(k) + \underline{f_y}^j(k)) \ge \overline{f_y}^i(k) + R^{i+j} + \frac{1}{2}(\overline{f_y}^j(k) - \underline{f_y}^j(k)). \end{aligned} \tag{6}$$

From (6) we again derive a set of constraints as in the rear part-to-rear part collision case.

As an alternative, we also formulated tighter front part-to-front part collision constraints by excluding the interval overlap of $[\underline{f_x}^i - R^i, \overline{f_x}^i + R^i]$ with $[\underline{f_x}^j - R^j, \overline{f_x}^j + R^j]$ and in y-direction vice versa. This formulation is slower as it needs more binary decision variables and does not provide huge benefits. In dense and coupled scenarios, (6) can lead to harsh braking or acceleration in front of narrow but still drivable passages due to the over-conservative approximation.

C. Joint Cost Function

As the objective of the optimization problem, we chose to minimize a weighted sum of individual cost functions per agent. This approach if referred to centralized planning in a model predictive control setting. We formulate the individual cost function term of agent A^i as

$$\begin{aligned} J^i = \sum_{k\in\mathcal{K}} \Big(& q_p^i (p_x^i(k) - p_{x,ref}^i(k))^2 + q_p^i (p_y^i(k) - p_{y,ref}^i(k))^2 \\ & + q_u^i u_x^i(k)^2 + q_u^i u_y^i(k)^2 \Big) \end{aligned} \tag{7}$$

with suitable weighting factors $q_\square$ consisting of terms for tracking the reference trajectory and terms for penalizing the jerk.

To leverage the interests of the agents, we introduce *scaling factors* $\lambda \in [0, 1]$ in the overall cost function. With these, we can push the optimization problem to generate egoistic, symmetric, or altruistic solutions. For each agent A^i, we define a scaling factor λ^i with the following properties. First,

$$\sum_{i\in\mathcal{A}} \lambda^i = 1. \tag{8}$$

The scaling factor of the ego agent λ^{ego} is chosen freely within the interval $[0, 1]$. We then set

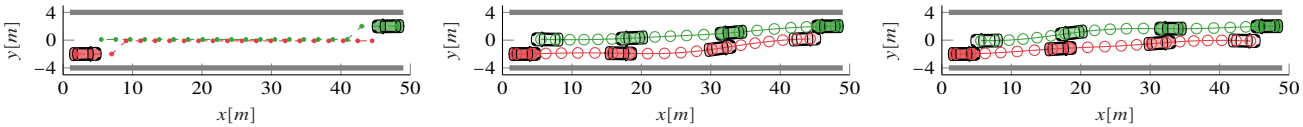
$$\lambda^i = \frac{1 - \lambda^{\text{ego}}}{|\mathcal{A}| - 1}\ \forall\ i \neq \text{ego}. \tag{9}$$

The intuitive explanation of (9) is that high values for λ^{ego} will lead to egoistic ego behavior, all values equal a symmetric solution, and for small values of λ^{ego} the ego agent will not enforce its own goals, only trying to fulfill the constraints.

The overall cost function is then defined as

$$J = \sum_{i\in\mathcal{A}} \lambda^i J^i + q_\xi \sum_{k\in\mathcal{K}} \sum_{i\in\mathcal{A},\ j\in\mathcal{A}\setminus i} (\xi_x^{ij}(k) + \xi_y^{ij}(k))^2 \tag{10}$$

with the second term penalizing high values of the slack variables and a weighting factor q_ξ.



(a) Reference tracks and initial conditions (b) Solution for $\lambda = 0.2$ (c) Solution for $\lambda = 0.6$

Fig. 3: A scenario with conflicting references. The optimized solution is leveraged with the cooperation factor λ.

D. Receding Horizon Formulation of the MIQP

We execute the algorithm in a receding horizon fashion and, therefore, only execute the first step of the optimized trajectory. The instance of the MIQP model solved per time step is formulated as in our previous work [12] sketched in Section IV-A, the constraints here are calculated individually for each agent. The model is enhanced with the agent-to-agent collision avoidance constraints introduced in Section IV-B. We minimize the joint cost function (10) defined in Section IV-C. As initial conditions, we set the states of each agent and the current region of each agent. Adding the latter helps to avoid infeasible problems on the region boundaries. In such cases, where the initial region is possibly ambiguous due to numerical inaccuracies, we start the optimization for both possible regions.

Due to the separation of x- and y-directions, absolute terms such as a reference path or the velocity along this path, cannot be included directly in the cost function. As we still consider it desirable to track a reference speed along a reference path, we compute the trajectory reference, a sequence of x and y coordinates along the discrete time k, from path and speed reference values in a preprocessing step.

To speed up the solution computation and to enforce finding consistent solutions close to the solution of the previous step, we warm-start the MIQP solver using the solution from the previous receding horizon instance [15]. From this previous solution, we use the decision variables from step two on and perform a simple extrapolation to initialize the variables at the last time step which introduces only a neglectable overhead regarding computation time. We leave a more advanced strategy, such as warm-starting the cuts or the branch-and-bound tree, to future work.

V. Evaluation

We demonstrate both the implications of the introduced cooperation factor λ in the joint cost function and the agent-to-agent collision constraint including the soft constraint term to account for model inaccuracies in two simulated scenarios. The algorithms are written in Mathworks MATLAB and the generated MIQP is solved using IBM Cplex.

A. Levels of Cooperation in a Negotiation Scenario

We evaluate the effect of the cooperation factor λ in a negotiation scenario with two agents A^1 and A^2. The scenario is fully symmetric with two vehicles placed on a two-lane road in oncoming direction with road boundaries. Both reference lines do not track the lane centers but the road center. This yields conflicting goals (Fig. 3a). The solution can be balanced to favor one or the other agent. The vehicle trajectories change as λ changes (Fig. 3b and Fig. 3c).

TABLE II: Quantitative evaluation of the negotiation scenario. We compare the overall distance to the reference, the time the reference is reached, and the contributions to the cost function of each agent.

λ	Dist. A^1	Dist. A^2	Idx A^1	Idx A^2	Cost A^1	Cost A^2
0	54.684	19.645	-	11	3.7452	5006.2
0.1	33.872	20.181	18	11	899.19	4568.5
0.2	30.788	21.362	17	12	1552.6	4183.6
0.3	28.937	22.543	17	14	2114.1	3796.1
0.4	28.383	22.894	17	14	2731.3	3305.4
0.5	22.98	28.297	14	17	2773	3380.1
0.6	22.884	28.383	14	17	3304.4	2731.3
0.7	22.532	28.929	14	17	3795.5	2113.2
0.8	21.364	30.772	12	17	4183.5	1551.5
0.9	20.168	33.865	11	18	4568.3	898.88
1	19.646	54.666	11	-	5006.2	3.699

In Table II, we qualitatively show the effect of varying λ. We analyze a single run of the algorithm. By Dist. $A^{\square}$ we denote the accumulated distance over all N time steps in meters from the solution trajectory to the reference for the respective agent. The column Idx $A^{\square}$ indicates at which time index the respective agent has reached the reference trajectory. We also state the contribution of each agent to the global cost function, denoted by Cost $A^{\square}$. All three metrics show the same trend; by varying λ the respective agent is favored. We observe that for $\lambda \approx 0.5$ all metrics are balanced, but also with a strong favor of one agent still, valid solutions are computed.

B. Competitive Racing

We evaluate our algorithm in an autonomous vehicle racing scenario, in which dense interactions with other agents

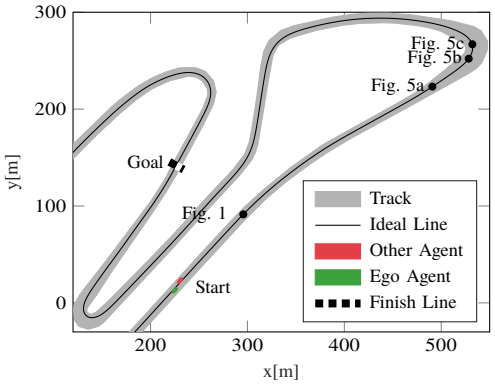
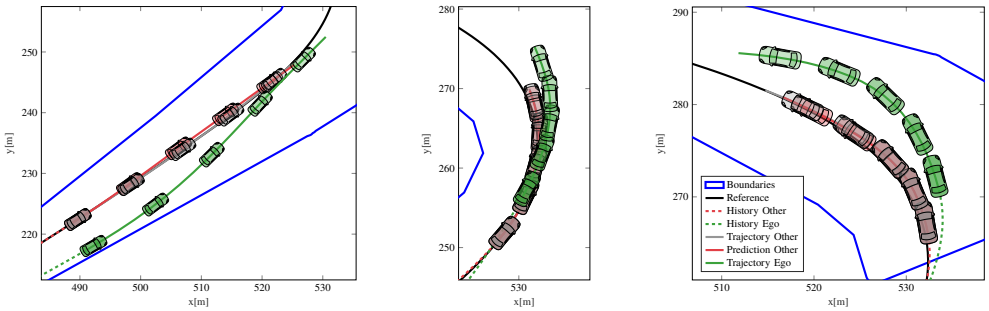


Fig. 4: Setup of the racing scenario. The start points of scenes depicted in the other figures are located on the track.



(a) $t = 27s$ (b) $t = 31s$ (c) $t = 34s$

Fig. 5: The ego agent A^1 overtaking in the first curve with cooperation factor $\lambda = 0.3$. Fig. 1 depicts $t = 9s$.

usually occur. The goals of each agent are conflicting as each agent wants to win the race. Besides the observations of other agents no communication is involved. The uncertainty in the intent of other agents is neglectable compared with road traffic in this setting. We assume the optimal line on the racetrack to be known and equal for all agents. Therefore, each agent competes to stay on this ideal line, which we use as a reference path. In the curves, we limit the maximum velocity with respect to the maximum lateral acceleration possible for the vehicle model.

We use a racing track of Berlin by Heilmeier *et al.* [16] with the provided ideal line and track boundaries (Fig. 4). We naively triangulate the track boundaries and greedily merge the triangles to get the convex approximation of the environment. We define the finish line after 1177 m. The ego agent A^1 starts with a disadvantage but has a slightly higher top speed than the other agent A^2, so it can eventually overtake and win the race. The ego agent A^1 uses our proposed multi-agent MIQP method to plan its behavior and trajectory, whereas the other agent A^2 only tracks the reference line with respect to its kinematic constraints without considering the ego agent. Table III states the scenario parameters.

In the overtaking scene in Fig. 1 and Fig. 5, we show exemplary for $\lambda = 0.3$ how A^1 finally makes use of its higher top speed. We observe, that even with a very inaccurate model of A^2 (Fig. 1), A^1 can safely catch up (Fig. 5a), finally overtake (Fig. 5b), and keep the leading position even though it has to take a wider curve due to its higher speed (Fig. 5c).

In Fig. 6, we analyze how different scaling factors λ lead to different agent behavior by tracking the lap time of both agents. As a baseline, we simulate each agent alone on the racetrack. If A^1 ignores the predicted motion and intention of A^2 ($\lambda > 0.85$), it drives too aggressive and eventually provokes a crash of both vehicles. In the other extreme ($\lambda < 0.1$), A^1 ignores its own goal, which leads to very passive behavior. It cannot successfully overtake even though it could accelerate to higher top speed. For scaling factors in between, we observe that depending on the λ, A^1 sooner or later successfully overtakes and wins the race. Hence, with a balanced scaling factor A^1 can achieve its own goal to drive at higher top speed, also taking into account the intent of A^2 to stay on the ideal line at a lower speed.

TABLE III: Parameter values of the racing scenario.

$v^1_{des.}$	$v^2_{des.}$	q^i_p	q^i_u	q_ξ	D	Δt	N
$12m/s$	$14m/s$	$10\lambda^i$	$0.5\lambda^i$	30	$3m$	$0.2s$	20

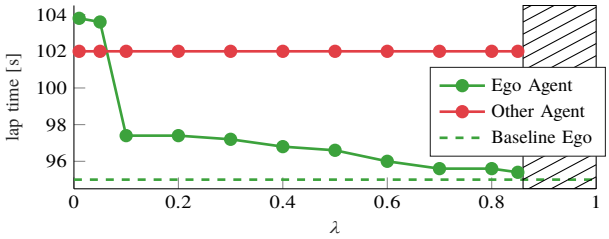
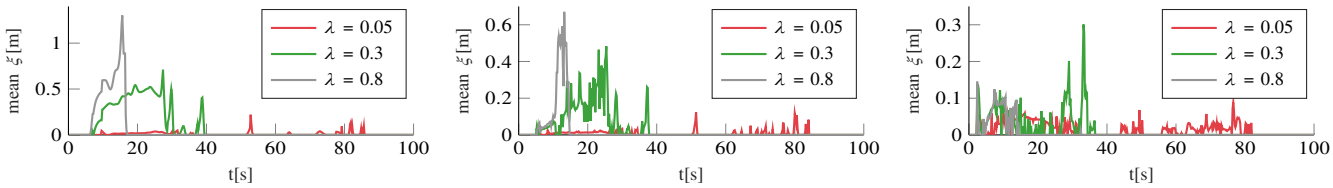


Fig. 6: How can the ego agent A^1 win the race? A^1 incorporates the motion of A^2. It looses the race if parameterized too conservative ($\lambda < 0.1$) and overtakes the sooner the more aggressive λ is chosen. If the interests of A^2 are ignored, eventually a crash occurs while overtaking (hatched area, $\lambda > 0.85$).

In Fig. 7, we analyze the effect of the scaling factor λ on the interaction of the agents and the safety distance the ego agent A^1 is willing to keep. If A^1 behaves aggressively, it implicitly models, that A^2 will make room, which it will not. This results in a high prediction error for high values of λ which is compensated by the slack terms. We observe that more slack is used to mitigate immanent collisions at the beginning of the planning horizon (Fig. 7a). At the end of the horizon, the magnitude of slack used is lower as the optimizer has cheaper alternatives to avoid collisions (changing position or jerk) even though the absolute errors from a non-ideal prediction is higher (Fig. 7c). We show three different values of λ representing three different behaviors of A^1; early overtaking ($\lambda = 0.8$, until $t = 17s$), late overtaking ($\lambda = 0.3$, until $t = 40s$, also see Fig. 5), and no overtaking ($\lambda = 0.05$). As soon as the overtaking maneuver has been performed successfully, the slack costs tend to zero, as the safety distance can trivially be fulfilled. In the edge case $\lambda = 0.05$, A^1 behaves very passive and always tries to keep the safety distance big. In the other extreme of $\lambda = 0.9$, A^1 behaves too aggressive, accepts a high prediction error that cannot be compensated by the slack terms and finally provokes a crash. For $\lambda = 0.8$, A^1 performs an aggressive, but still safe, overtaking maneuver, whereas for $\lambda = 0.3$ it



(a) Horizon index $k = 2$ (b) Horizon index $k = 10$ (c) Horizon index $k = 20$

Fig. 7: The effect of soft constraints coping with prediction errors. At the respective horizon index, we depict the mean safety distance compensated by the slack variables ξ for different factors λ.

initiates the overtaking late as more drivable area is available. Hence, with appropriate parameter selection, our algorithm can cope with not completely known cost functions of other agents and the driving style of the ego agent, including its willingness to take risks can be adopted.

VI. Conclusion and Future Work

We introduced a novel linear differential game formulation for the multi-agent behavior planning problem leveraging the interests of all agents. Using linear constraints for checking collision between agents, we formulate a mixed-integer quadratic program and obtain an optimal solution. The introduction of slack variables to the collision constraints makes our method robust against inaccuracies of the modeled future motion of other agents.

We first studied the impact of an altruistic, cooperative, and egoistic behavior setting on the outcome of a symmetric scenario with conflicting goals of both agents. We then demonstrated the effectiveness of our method in a competitive autonomous vehicle racing scenario, where we introduced inaccuracies between the true and modeled behavior of the other agent. In this highly interactive scenario that poses high requirements on the receding horizon replanning scheme, we showed that the proposed multi-agent planning approach correctly satisfies all model constraints and also copes with inaccurate agent models. Furthermore, the introduced cooperation factor in the joint cost function allows leveraging the goals of either agent. In combination with our previous work [12], which proved correctness in the model of the vehicle kinematics and the obstacle avoidance, we have successfully demonstrated the feasibility in simulation.

As a next step, we plan to apply the formulation in real-road driving scenarios using the institute's research vehicle [17]. For this to be feasible regarding computation time, we need to find a trade-off which agents to denote as interacting and which others to predict maneuvers for solely. The approach is applicable for an arbitrary number of agents, but an analysis on the computational scalability has to be conducted. Also, the assumption on perfect observation of other agents has to be weakened by introducing suitable observers and only optimizing within the sensing radius. Uncertainty measures from the perception pipeline shall be used to parametrize the soft constraints online. Furthermore, modeling traffic rules as logical constraints, which pose constraints to the possible behavior to choose from, shall be elaborated to better approximate future behaviors.

Acknowledgment

This research was partly funded by the Bavarian Ministry of Economic Affairs, Regional Development and Energy, project Dependable AI and supported by the Intel Collaborative Research Institute - Safe Automated Vehicles.

References

[1] W. Schwarting, J. Alonso-Mora, and D. Rus, "Planning and Decision-Making for Autonomous Vehicles," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, no. 1, 2018.

[2] S. Ulbrich, S. Grossjohann, C. Appelt, *et al.*, "Structuring Cooperative Behavior Planning Implementations for Automated Driving," in *IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2015.

[3] T. Kessler and A. Knoll, "Cooperative Multi-Vehicle Behavior Coordination for Autonomous Driving," in *IEEE Intelligent Vehicles Symposium (IV)*, 2019.

[4] D. Lenz, T. Kessler, and A. Knoll, "Tactical Cooperative Planning for Autonomous Vehicles using Monte-Carlo Tree Search," in *IEEE Intelligent Vehicles Symposium (IV)*, 2016.

[5] M. Bahram, A. Lawitzky, J. Friedrichs, *et al.*, "A Game-Theoretic Approach to Replanning-Aware Interactive Scene Prediction and Planning," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 6, 2016.

[6] A. Liniger and J. Lygeros, "A Noncooperative Game Approach to Autonomous Racing," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 3, 2020. arXiv: 1712.03913.

[7] W. Schwarting, A. Pierson, S. Karaman, *et al.*, "Stochastic Dynamic Games in Belief Space," 2019. arXiv: 1909.06963.

[8] F. Fabiani and S. Grammatico, "Multi-Vehicle Automated Driving as a Generalized Mixed-Integer Potential Game," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, 2020.

[9] J. Eilbrecht and O. Stursberg, "Cooperative Driving using a Hierarchy of Mixed-Integer Programming and Tracking Control," in *IEEE Intelligent Vehicles Symposium (IV)*, 2017.

[10] S. Manzinger and M. Althoff, "Tactical Decision Making for Cooperative Vehicles Using Reachable Sets," in *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2018.

[11] C. Frese and J. Beyerer, "A Comparison of Motion Planning Algorithms for Cooperative Collision Avoidance of Multiple Cognitive Automobiles," in *IEEE Intelligent Vehicles Symposium (IV)*, 2011.

[12] K. Esterle, T. Kessler, and A. Knoll, "Optimal Behavior Planning for Autonomous Driving: A Generic Mixed-Integer Formulation," in *IEEE Intelligent Vehicles Symposium (IV)*, 2020. arXiv: 2003.13312.

[13] S. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.

[14] B. Gutjahr, L. Gröll, and M. Werling, "Lateral Vehicle Trajectory Optimization Using Constrained Linear Time-Varying MPC," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 6, 2017.

[15] T. Marcucci and R. Tedrake, "Warm Start of Mixed-Integer Programs for Model Predictive Control of Hybrid Systems," *IEEE Transactions on Automatic Control*, 2020.

[16] A. Heilmeier, A. Wischnewski, L. Hermansdorfer, *et al.*, "Minimum Curvature Trajectory Planning and Control for an Autonomous Race Car," *Vehicle System Dynamics*, 2019.

[17] T. Kessler, J. Bernhard, M. Buechel, *et al.*, "Bridging the Gap between Open Source Software and Vehicle Hardware for Autonomous Driving," in *IEEE Intelligent Vehicles Symposium (IV)*, 2019.