

# Hierarchical Time-Optimal Planning for Multi-Vehicle Racing\*

Georg Jank, Matthias Rowold, and Boris Lohmann

**Abstract**—This paper presents a hierarchical planning algorithm for racing with multiple opponents. The two-stage approach consists of a high-level behavioral planning step and a low-level optimization step. By combining discrete and continuous planning methods, our algorithm encourages global time optimality without being limited by coarse discretization. In the behavioral planning step, the fastest behavior is determined with a low-resolution spatio-temporal visibility graph. Based on the selected behavior, we calculate maneuver envelopes that are subsequently applied as constraints in a time-optimal control problem. The performance of our method is comparable to a parallel approach that selects the fastest trajectory from multiple optimizations with different behavior classes. However, our algorithm can be executed on a single core. This significantly reduces computational requirements, especially when multiple opponents are involved. Therefore, the proposed method is an efficient and practical solution for real-time multi-vehicle racing scenarios.

## I. INTRODUCTION

Planning trajectories in environments with dynamic obstacles is a major task in autonomous driving. Although approaches for traffic scenarios and racing can be similar, high speeds, small distances, and different rules pose a unique challenge in competitive driving on race tracks (like the Indy Autonomous Challenge). Trajectory planning in this environment requires rapid solving of non-convex optimization problems to generate time-optimal behavior (e.g. left or right overtake) with a corresponding feasible trajectory.

The majority of recent planning approaches for racing solve the behavior and trajectory generation problem in one step by selecting the cost-minimum option from a finite number of generated trajectories [1]–[3]. These methods are not prone to local optima, as they cover a large region of the search space. However, they only find discrete-optimal solutions, as they do not explore all possible trajectories. We call them discrete methods in the following. Numerical optimization-based methods, on the other hand, solve an optimal control problem (OCP) with only the time or progress along a curve being discretized. Thus, they are often referred to as continuous methods. As the control problem is non-convex, they converge to different local optima, depending on the initialization. One way to consider the non-convexity is to solve multiple OCPs in parallel, one for each behavior class. However, this does not scale well for multiple opponents and relies on parallel processing capabilities to achieve low computation times.

\*This work was not supported by any organization

All authors are with the Chair of Automatic Control, Department of Mechanical Engineering, TUM School of Engineering and Design, Technical University of Munich, 85748 Garching, Germany [georg.jank@tum.de](mailto:georg.jank@tum.de)

For rapid planning in an environment with multiple opponents, we propose a hierarchical planning approach that uses a spatio-temporal visibility graph to determine a high-level behavior and set the constraints for a low-level numerical optimization. In essence, we adopt discrete methods for exploration and continuous methods for exploitation, thereby combining the strengths of both approaches.

## II. RELATED WORK

Discrete planning methods generate and compare a finite number of candidate trajectories. There are two main subcategories of discrete planning approaches: sampling-based and graph search methods [4]. Sampling-based methods, using a rapidly-exploring random tree (RRT) [5], generate trajectory candidates randomly with forward dynamics. These are checked for feasibility and ranked to find the discrete-optimal trajectory. Other sampling-based approaches, applied in racing, sample jerk-minimal splines [1], [3]. A major disadvantage of sampling-based methods is the large number of candidates required to plan complex driving maneuvers [6].

Graph search methods aim to reduce the number of trajectory candidates by creating a graph of feasible trajectory segments called edges. A graph search then determines the cost-minimal sequence of edges. In path-velocity decomposition [7], trivial overtaking maneuvers are planned by calculating a collision-free velocity profile on an optimal path derived from a spatial graph. Even though such approaches have been applied in racing [8], they are not time-optimal, as they do not fully capture the spatio-temporal character of the problem. Directly considering dynamic obstacles in the graph leads to spatio-temporal graphs [2]. However, this requires at least one additional dimension (time, velocity, or both). Therefore, the discretization must be kept coarse to mitigate the curse of dimensionality.

Continuous methods only discretize the time or progress along a curve and solve an OCP numerically. As they converge to continuous local optima, they have become a common choice for trajectory planning in autonomous motor-sport [9]–[14]. Due to the non-convexity of most planning problems in racing, different initial guesses can lead to different local optima [14]. To find the global optimum, a common approach is to solve multiple OCP in parallel, one for each homotopy class, i.e. overtaking behavior [10], [12]. The results are then compared in search of the progress-maximizing solution. This approach increases the chances of finding the global optimum at the cost of computational complexity. A different approach that reduces online computational effort is to determine overtaking with a policy learned from offline simulations [11]. While this method is

fast, it is not versatile because the calculated policy is only valid for a specific track.

Lim et al. [15] propose a hierarchical planning approach for traffic scenarios. The behavior is determined with a spatio-temporal graph search, and the solution is used to initialize an OCP. This method combines the ability of discrete methods to find solutions close to the global optimum with the precision of trajectories calculated with continuous methods. However, the algorithm is only viable with a low resolution of the graph, resulting in too conservative behaviors for racing.

There are several ways to enforce overtaking behavior in numerical optimization algorithms. Some authors suggest initializing the OCP with a trajectory estimate, following the behavior [9], [10], [12]. Other methods convert the non-convex OCP into a convex subproblem by limiting motion to a maneuver envelope so that the behavior of the planned trajectory is more predictable [11], [16], [17]. This is especially important in racing, where following the optimal behavior is critical.

### A. Contributions

We introduce a hierarchical planning method that extends the local racing line algorithm in [13] for multi-vehicle scenarios. Inspired by [15], we combine discrete high-level behavioral planning with low-level numerical optimization. This reduces computational complexity compared to [12] and [10] and improves flexibility compared to [11]. The main contributions to the hierarchical approach are as follows:

- We propose a behavioral planning step based on spatio-temporal graphs. In contrast to [15], temporal planning precedes spatial planning. Progress variants, derived from the previous planning iteration, determine the geometry of spatial planning problems that are solved with low-resolution visibility graphs.
- We adapt the constraints and cost function of the time-optimal control problem in [13] to generate a feasible trajectory for the generated high-level behavior.
- We perform a monte carlo simulation to compare our approach with parallel optimization-based methods and naive overtaking strategies. We analyze the results regarding computation time and driving performance.

## III. METHODOLOGY

Our approach operates in two modes shown in Figure 1: (1) Without any opponents in the planning horizon, the trajectory is generated according to [13]. A time-optimal control problem with a point mass model, constrained by gg-diagrams, is solved for the upcoming track section. In Sections III-A and III-B, we will briefly summarize the used track and vehicle model.

(2) When opponents are present, the first step is to make a behavioral decision, whether to pass opponents on the left or right and to define a corresponding maneuver envelope. These processes are explained in Sections III-C and III-D. The second step, described in Section III-E, is to solve the

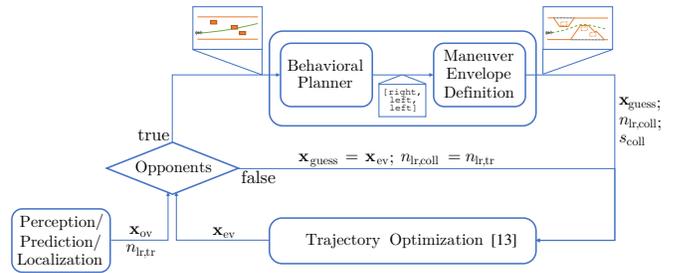


Fig. 1. Overview of the hierarchical planning approach.

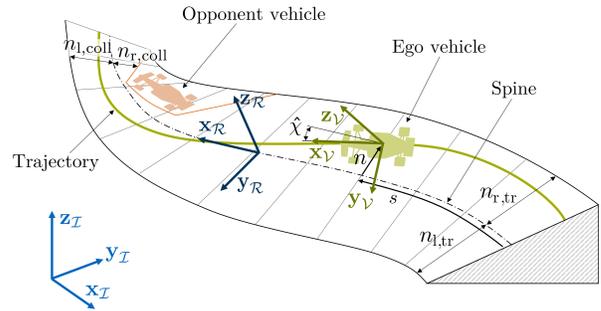


Fig. 2. 3D track with road frame  $\mathcal{R}$  and velocity frame  $\mathcal{V}$ .

time-optimal OCP with constraints adapted to comply with the determined maneuver envelope.

### A. Track Model

We use the curve-ribbon approach for modeling three-dimensional (3D) tracks, presented in [18]. The road frame  $\mathcal{R}$  moves along a 3D reference curve, called the spine. It defines the road surface, as shown in Figure 2. The arc length along the spine is denoted as  $s$ , while  $n$  is the lateral displacement in the direction of the  $y$ -axis of  $\mathcal{R}$ . The rotation rate of  $\mathcal{R}$  with respect to arc length  $s$  is expressed as angular velocity  ${}_{\mathcal{R}}\Omega_{\mathcal{R}} = [\Omega_x \ \Omega_y \ \Omega_z]^T$  in the  $\mathcal{R}$ -frame. For a detailed description of the 3D track representation, we refer to [18].

### B. Vehicle Model

Following [13], we use a low-dimensional point mass model to describe the dynamics of the vehicle. The state  $\mathbf{x}$  is defined as

$$\mathbf{x} = [V \ n \ \hat{\chi} \ \hat{a}_x \ \hat{a}_y]^T, \quad (1)$$

where  $V$  is the velocity and  $\hat{\chi}$  is the orientation of the velocity-alinged frame  $\mathcal{V}$  relative to the road frame  $\mathcal{R}$ . The longitudinal and lateral accelerations of  $\mathcal{V}$  are given by  $\hat{a}_x$  and  $\hat{a}_y$ , respectively. The accelerations are constrained by gg-diagrams according to [13]. The longitudinal and lateral jerks  $\hat{j}_x$  and  $\hat{j}_y$  form the input vector

$$\mathbf{u} = [\hat{j}_x \ \hat{j}_y]^T. \quad (2)$$

With the vertical velocity  $w$  and the angular velocity of  $\mathcal{V}$  with respect to time  ${}_{\mathcal{V}}\omega_{\mathcal{V}} = [\hat{\omega}_x \ \hat{\omega}_y \ \hat{\omega}_z]^T$ , the dynamics

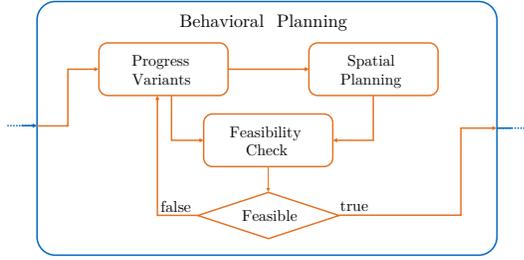


Fig. 3. Overview of the behavioral planning algorithm.

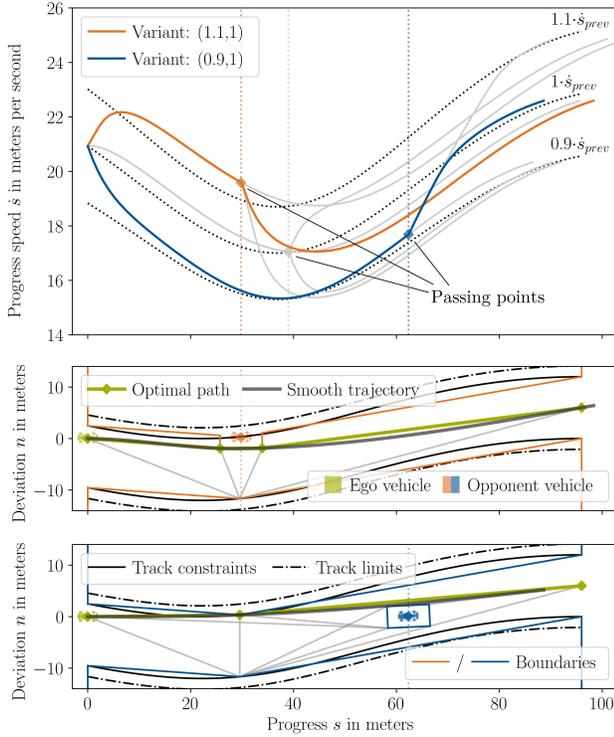


Fig. 4. Behavioral planning for an example maneuver with a single opponent. The top figure shows the generation of progress variants, while the middle and bottom figures depict the visibility graphs for the variants (1.1,1) and (0.9,1).

are described by

$$\dot{\mathbf{x}} = \frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}, \mathbf{u}) = \begin{bmatrix} \hat{a}_x - w\hat{\omega}_y \\ V \sin(\hat{\chi}) \\ \frac{\hat{a}_y + w\hat{\omega}_x}{V} - \Omega_z \hat{s} \\ \hat{j}_x \\ \hat{j}_y \end{bmatrix}. \quad (3)$$

### C. Behavioral Planning

Given the predicted motion of the opponent vehicles, the behavioral planning step approximates an optimal overtaking trajectory and selects the fastest sequence of left or right passing decisions. Since the times and positions of overtakes depend on the progress of the ego vehicle, this is a spatio-temporal problem. We solve this problem by first sampling progress variants, second finding the optimal path for a given

variant, and third checking the feasibility of the resulting trajectory. Following this order allows for the creation of low-resolution visibility graphs that take advantage of the problem geometry.

1) *Progress Variants*: Progress variants are generated by following different set speed profiles  $\dot{s}_{\text{set}}$ , based on the optimal trajectory of the previous planning iteration  $\mathbf{x}_{\text{prev}}$ . To do this, we modulate acceleration with a feedback controller  $\ddot{s} = K(\dot{s}_{\text{set}} - \dot{s})$ . Following a certain speed profile  $\dot{s}_{\text{set}}$  determines the  $s$ -coordinate where the next opponent vehicle is passed. At these passing points, the speed profiles branch out by switching to different set speed profiles. The passing points for a single opponent and set speed profiles  $\dot{s}_{\text{set}} \in \{0.9 \cdot \dot{s}_{\text{prev}}, 1 \cdot \dot{s}_{\text{prev}}, 1.1 \cdot \dot{s}_{\text{prev}}\}$  are shown in the top diagram of Figure 4. In the same diagram, we highlight two exemplary progress variants (1, 1.1) and (0.9, 1). The first variant follows  $1.1\dot{s}_{\text{prev}}$  at the start of the maneuver and switches to  $1\dot{s}_{\text{prev}}$  after the overtake, while the second one goes from  $0.9\dot{s}_{\text{prev}}$  to  $1\dot{s}_{\text{prev}}$ . For multiple opponents, the aforementioned procedure can quickly result in a large number of progress variants. With three speed profiles and  $N$  opponents,  $3^{N+1}$  variants are possible. Performing spatial planning, as described in Section III-C.2, for all variants would be too computationally complex. Therefore, we generate the progress variants as needed, beginning with the fastest variant. If the spatial planning step can generate a feasible trajectory for the current variant, the trajectory and corresponding behavior are applied to the numerical optimization. Otherwise, we continue with the next fastest variant. This iterative procedure is visualized in Figure 3 and promotes finding the global time-optimal solution. More details on the feasibility checks are given in Section III-C.3.

2) *Spatial Planning*: With the passing points of the considered progress variant, a spatial graph can be generated. We utilize visibility graphs. These are undirected graphs, connecting all vertices of obstacles with straight edges that do not cross an obstacle [19]. Originally, they were developed to find the shortest collision-free path. Compared to the spatio-temporal lattice with fixed nodes [15], the discretization, based on the corner points of moving obstacle polygons, allows for the generation of short and direct path candidates with a small number of nodes. Considering vehicle dimensions and safety distances, we virtually expand the track boundaries and opponent polygons to avoid collisions when the center point of the ego vehicle is within bounds. The visibility graphs for the progress variants (1, 1.1) and (0.9, 1) are shown in the two bottom diagrams of Figure 4.

An A\* search determines the optimal path to minimize the total travel distance  $\sum_i d_i$  and angle deviation  $\sum_i |\hat{\chi}_i|$  relative to the spine:  $\min_i (w_d d_i + w_\chi |\hat{\chi}_i|)$ . The search is guided by a heuristic function  $h(P)$ , based on the length  $d_{\overline{PD}}$  and angle deviation  $|\hat{\chi}_{\overline{PD}}|$  of a virtual edge  $\overline{PD}$  connecting the current point  $P$  to the destination  $D$ :  $h(P) = w_d d_{\overline{PD}} + w_\chi |\hat{\chi}_{\overline{PD}}|$ . By discouraging long and weaving paths, the goal is to predict which path is most likely to be feasible for the given progress variant. We increase the speed of the

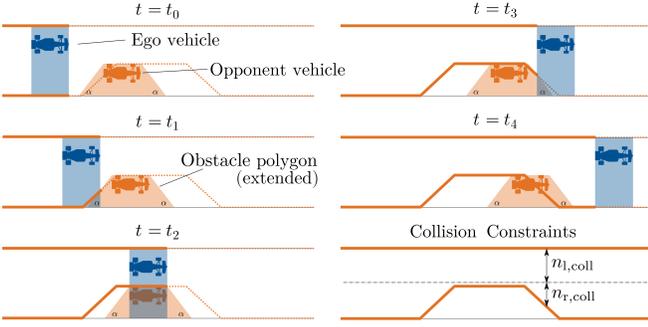


Fig. 5. Generation of maneuver envelopes.

search algorithm by applying the following simplifications to reduce the number of nodes in the graph: (1) With the help of the Ramer–Douglas–Peucker algorithm [20], we reduce the number of boundary points to a subset of points that approximates the shape. (2) We remove the boundary nodes at the start and end of the planning horizon, as the vehicle would have to drive perpendicular to the spine or in reverse track direction to reach them.

3) *Feasibility Check*: Spatial planning with visibility graphs results in non-continuous curvature profiles, so the unprocessed paths are not feasible. To confirm the suitability of a path and its corresponding overtaking behavior, a cubic spline  $f_{\text{spline}}(s)$  is placed through the path, as seen in Figure 4. The smoother path candidate  $n_{\text{cand}} = f_{\text{spline}}(s_{\text{cand}}(t))$  is then combined with the considered progress variant  $\dot{s}_{\text{cand}}(t)$  to form the trajectory candidate

$$\mathbf{x}_{\text{cand}} = \begin{bmatrix} V_{\text{cand}} \\ n_{\text{cand}} \\ \hat{\chi}_{\text{cand}} \\ \hat{a}_{x,\text{cand}} \\ \hat{a}_{y,\text{cand}} \end{bmatrix} = \begin{bmatrix} \frac{\dot{s}_{\text{cand}}(1-n_{\text{cand}}\Omega_z)}{\cos \hat{\chi}} \\ n_{\text{cand}} \\ \arctan f'_{\text{spline}}(s_{\text{cand}}) \\ \dot{V} \\ V(\omega_z + \hat{\chi}) \end{bmatrix}. \quad (4)$$

If the accelerations from the trajectory  $\mathbf{x}_{\text{cand}}$  lie within the gg-diagrams, behavioral planning finishes with the trajectory estimate  $\mathbf{x}_{\text{guess}} = \mathbf{x}_{\text{cand}}$  and its corresponding behavior. Otherwise, the next slower progress variant is examined.

#### D. Maneuver Envelope Definition

The maneuver envelope should force the solution of the OCP with initialization  $\mathbf{x}_{\text{guess}} = \mathbf{x}_{\text{cand}}$  to remain in the previously determined optimal behavior class.

The maneuver envelopes are formed by extending obstacle polygons of the opponents to cover the side of the track where overtaking is suboptimal according to the behavioral planning step. As the vehicle travels along the planning horizon, the resulting spatio-temporal obstacle constraints form a narrowed driving corridor, as depicted in Figure 5. We reduce complexity by combining all obstacle constraints into collision constraints that describe this narrowed driving space. This is achieved by sampling the lateral restriction  $n \in [n_{r,\text{coll}}(s), n_{l,\text{coll}}(s)]$  for the initial guess  $\mathbf{x}_{\text{guess}}$ .

While the complexity of the OCP is significantly reduced, information gets lost when spatio-temporal constraints are

reduced to spatial constraints. As the constraints now only depend on the distance, they can influence the vehicle speed solely through the feasible curvatures in the narrowed driving space. To make the vehicle slow down when the gap for overtaking is too small, we re-introduce the spatio-temporal component by adding a constraint on vehicle progress  $s < s_{\text{coll}} + V_{\text{coll}}t$ . This longitudinal constraint acts like a wall moving at the average speed of the obstacle.

#### E. Optimal Control Problem

Following [13], the second step of our hierarchical approach solves an OCP parametrized by  $s$  for a constant spatial planning horizon  $s \in [s_0, s_e]$ . By using numerical optimization, we can calculate fast trajectories that are not limited by discretization. The cost function (5a) consists of three terms: a time optimality term, a term that smooths the acceleration profile by minimizing jerk, and a slack term that ensures that the soft constraints on velocity and vehicle position are fulfilled. The OCP is defined as

$$\min_{\mathbf{x}, \mathbf{u}} \int_{s_0}^{s_e} \frac{1}{\dot{s}} + \mathbf{u}^\top \mathbf{R} \mathbf{u} + \epsilon^\top \mathbf{S} \epsilon \, ds \quad (5a)$$

$$\text{s.t. } \mathbf{x}' = \mathbf{f}(\mathbf{x}, \mathbf{u}) \frac{1}{\dot{s}} \quad (5b)$$

$$V - \epsilon_V \leq V_{\text{max}} \text{ with } \epsilon_V \geq 0 \quad (5c)$$

$$(9a), (9b), (9c) \text{ in [13]} \quad (5d)$$

$$n_{r,\text{tr}} + d_s \leq n \leq n_{l,\text{tr}} - d_s \quad (5e)$$

$$-\frac{\pi}{2} \leq \hat{\chi} \leq \frac{\pi}{2} \quad (5f)$$

$$n - \epsilon_{n_{l,\text{coll}}} + d_s \leq n_{l,\text{coll}} \text{ with } \epsilon_{n_{l,\text{coll}}} \geq 0 \quad (5g)$$

$$n + \epsilon_{n_{r,\text{coll}}} - d_s \geq n_{r,\text{coll}} \text{ with } \epsilon_{n_{r,\text{coll}}} \geq 0 \quad (5h)$$

$$s - \epsilon_{s,\text{coll}} \leq s_{\text{coll}} \text{ with } \epsilon_{s,\text{coll}} \geq 0 \quad (5i)$$

with

$$\mathbf{R} = \begin{bmatrix} w_{j,x} & 0 \\ 0 & w_{j,y} \end{bmatrix},$$

$$\epsilon = [1 \quad \epsilon_V \quad \epsilon_{n_{l,\text{coll}}} \quad \epsilon_{n_{r,\text{coll}}} \quad \epsilon_{s_{\text{coll}}}]^\top,$$

$$\mathbf{S} = \begin{bmatrix} 0 & \frac{w_{\epsilon,V,1}}{2} & \frac{w_{\epsilon,n_{l,\text{coll}},1}}{2} & \frac{w_{\epsilon,n_{r,\text{coll}},1}}{2} & \frac{w_{\epsilon,s_{\text{coll}},1}}{2} \\ \frac{w_{\epsilon,V,1}}{2} & w_{\epsilon,V,2} & 0 & 0 & 0 \\ \frac{w_{\epsilon,n_{l,\text{coll}},1}}{2} & 0 & w_{\epsilon,n_{l,\text{coll}},2} & 0 & 0 \\ \frac{w_{\epsilon,n_{r,\text{coll}},1}}{2} & 0 & 0 & w_{\epsilon,n_{r,\text{coll}},2} & 0 \\ \frac{w_{\epsilon,s_{\text{coll}},1}}{2} & 0 & 0 & 0 & w_{\epsilon,s_{\text{coll}},2} \end{bmatrix}.$$

Constraint (5b) enforces the equations of motion in (3). Using the diamond interpolation method presented in [13], we limit the combined accelerations in (5d). The vehicle is kept a safety margin  $d_s$  away from the track boundaries in (5e). Inequality (5f) prevents driving in reversed track direction. The aforementioned restrictions are hard constraints that have to be satisfied for a solution to exist.

However, there are cases where such a strict definition of constraints might be disadvantageous regarding the robustness of the solver. E.g., the speed limit is not safety-critical and can be violated for short periods of time. The soft velocity constraint is realized by the slack variable  $\epsilon_V$  in (5c). Similarly, the maneuver envelopes from Section III-D

are realized as soft constraints with (5g)–(5h). With hard constraints, the result, if feasible, would be too conservative because the uncertainty of the opponent’s prediction increases with distance. Following [21], our slack variables have linear and quadratic terms in the cost function. These are realized by the matrix  $S$ . If the initialization of the OCP violates one of the soft constraints  $x_0 \not\leq x_{\max}$ , the corresponding slack variable  $\epsilon_x$  is initialized with the value of the excess  $\epsilon_{x,0} = x_0 - x_{\max}$ . Within and between the planning steps, the violation is gradually decreased and eventually eliminated. The linear and quadratic weights  $w_{x,1}$ ,  $w_{x,2}$  determine how hard the violations are penalized and are therefore used for adjusting the softness of the constraints. This is especially useful for the collision constraints (5g)–(5i). Here, the slack weights  $w_{\epsilon,i_{\text{coll}},j}(s)$  for  $i \in \{n_l, n_r, s\}$  and  $j \in \{1, 2\}$  are defined as a function of progress with parameters  $w_{\epsilon,i_{\text{coll}},j,s_0}$  and  $w_{\epsilon,i_{\text{coll}},j,se}$  ( $w_{\epsilon,i_{\text{coll}},j,s_0} > w_{\epsilon,i_{\text{coll}},j,se}$ )

$$w_{\epsilon,i_{\text{coll}},j}(s) = w_{\epsilon,i_{\text{coll}},j,se} \left( \frac{w_{\epsilon,i_{\text{coll}},j,s_0}}{w_{\epsilon,i_{\text{coll}},j,se}} \right)^{\frac{s_e - s}{s_e - s_0}}. \quad (6)$$

Large slack weights close to the ego vehicle ( $s \approx s_0$ ) reduce the likelihood of collisions. Meanwhile, low weights at the end of the planning horizon make the solver more stable in the presence of large and sudden changes in the predicted vehicle position. For the velocity constraint (5c), we use constant slack weights  $w_{\epsilon,V,1}$  and  $w_{\epsilon,V,2}$ .

As long as the behavior, determined by the high-level planning step in Section III-C, remains the same, we initialize the OCP with the solution of the previous optimization  $\mathbf{x}_{\text{guess}} = \mathbf{x}_{\text{prev}}$ . If the behavior changes, the smoothed trajectory, passing the feasibility check in Section III-C.3, is used as a new initial guess for the OCP  $\mathbf{x}_{\text{guess}} = \mathbf{x}_{\text{cand}}$ .

#### IV. RESULTS

To validate and evaluate the hierarchical planning approach, we perform randomized simulations with three vehicles on the Modena race track. All simulations are calculated on an Intel Core i7-5600U CPU. The planning horizon is set to  $H = s_e - s_0 = 300$  m and progress variants are generated with the set speed factors  $\{0.5, 1, 2\}$ . The slack weights  $w_{\epsilon,n_{\text{coll}},1,s_0/se} = 50/25$ ,  $w_{\epsilon,n_{\text{coll}},2,s_0/se} = 5/2.5$ ,  $w_{\epsilon,s_{\text{coll}},1,s_0/se} = 20/10$  and  $w_{\epsilon,s_{\text{coll}},2,s_0/se} = 2/1$  are selected for collision-free overtaking. The definition of all other parameters was guided by [13]. At the beginning of each simulation, two opponent vehicles are positioned at random within a range of 120 m in front of the ego vehicle. Once the ego vehicle has overtaken both opponents and opened a gap of 50 m, the simulation is stopped. Each initial configuration is tested with the following four approaches:

- 1) The hierarchical approach presented in this paper
- 2) A pseudo-parallel optimization approach
- 3) Overtaking only on the left of the opponents
- 4) Overtaking only on the right of the opponents

The pseudo-parallel approach is based on the parallel optimization from Section II. For each behavior class, we specify a maneuver envelope and solve an OCP. To initialize

the different OCPs, we generate path candidates with cubic splines. These paths start at the current vehicle position, pass through points of the obstacle polygons and finish on the track spine at the end of the planning horizon. We initialize the OCP with  $\mathbf{x}_{\text{guess}} = \mathbf{x}_{\text{prev}}$  for the behavior class equal to the previous solution of the planning algorithm. For a comparison with our single-core hierarchical approach, we solve all OCPs sequentially. We name this single-core variant of parallel optimization pseudo-parallel optimization.

Exemplary overtakes for approaches 3) and 4) are shown in Figure 6. When overtaking on the left is specified, the vehicle passes on the outside of the first corner. For the overtake on the right, the vehicle accelerates less at the beginning to overtake when a gap opens up on the outside of the second turn. In this scenario, the left behavior class results in an earlier overtake compared to the right one.

Figure 7 shows the duration of the overtaking simulations for approaches 1)–4). Compared to the first two methods, the fixed behaviors produce significantly longer and more inconsistent overtaking times. This shows that there are multiple local optima and that it is beneficial to select the correct behavior before solving the OCP. The overtaking times from our hierarchical approach are similar to the pseudo-parallel approach that assesses all behavior classes in detail. Thus, we deduce that our proposed method selects the optimal behavior in the majority of cases.

The ego vehicle considers the vehicles within its planning horizon. To evaluate the scalability of the planning approaches 1)–4), we assess the calculation times for different numbers of opponents within the planning horizon. The results are shown in Figure 8. For the constant overtaking behaviors 3) and 4), the calculation times are similar and largely unaffected by the addition of opponents. For our hierarchical approach, there is a 0.6 s jump when introducing the first opponent. Adding a second opponent does not result in any further increase. Thus, computational complexity appears to scale well with the number of opponents.

Conversely, calculation time increases exponentially for the pseudo-parallel approach. Instead of planning in two steps, it optimizes a trajectory for every possible behavior combination. As there are  $2^N$  combinations for  $N$  opponents (e.g.  $N = 2 \rightarrow [l,l,r,r]$ ), the calculation time doubles for every added opponent. Going from zero to one opponent, the computation time increases even more, as only one of the OCPs is initialized with the previous solution when there are opponents. The other initial guesses are farther from their corresponding optima, and the OCPs take longer to converge.

#### V. CONCLUSION

In this paper, we introduced a hierarchical planning approach for racing with multiple opponent vehicles. We demonstrated that our approach with low-resolution visibility graphs for spatio-temporal planning is computationally efficient and has overtaking performance similar to parallel optimization. Therefore, the method is to be preferred when the computational resources are limited to a single core.

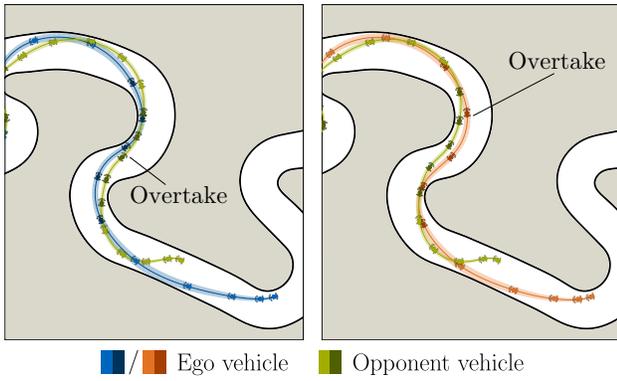


Fig. 6. Example for left overtaking behavior (left) and right overtaking behavior (right). The lines show the paths of the vehicles and line thickness indicates speed. The vehicle positions are plotted at intervals of  $\approx 1.2$  s.

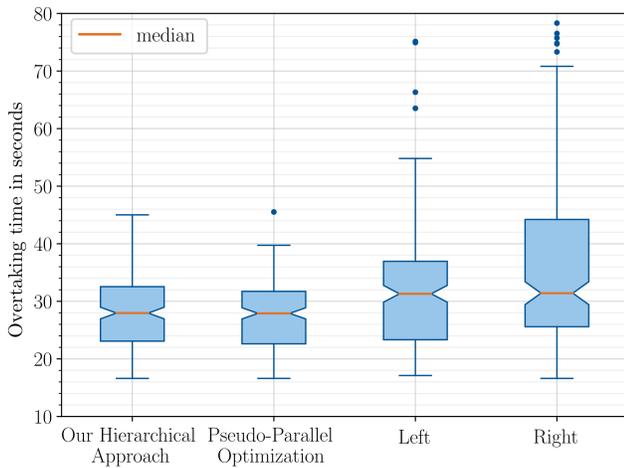


Fig. 7. Time required for overtaking two opponent vehicles based on 216 randomized overtaking simulations. The median overtaking performance of our hierarchical approach is similar to the pseudo parallel optimization. Compared to that, the fixed strategies generally result in slower overtakes.

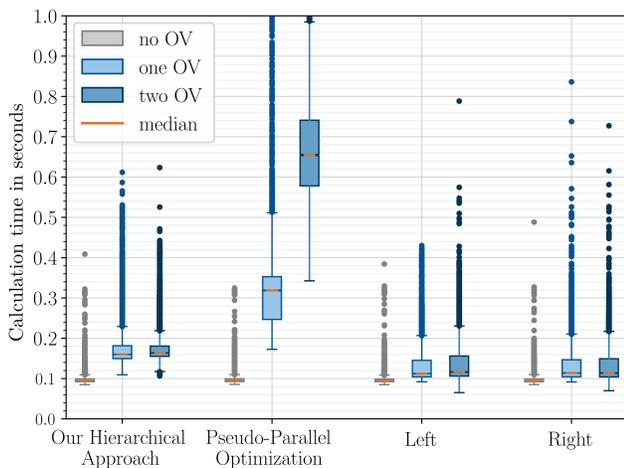


Fig. 8. Calculation time for different maneuvers and numbers of opponent vehicles (OV) based on 216 simulations with approximately 300 steps per simulation. The hierarchical approach shows slightly higher computation times than methods without behavioral planning. However, in contrast to the pseudo-parallel approach, the computation time does not increase exponentially with the number of opponents. Thus, it is more scalable.

## REFERENCES

- [1] A. Raji, A. Liniger, A. Giove, A. Toschi, N. Musiu, D. Morra, M. Verucchi, D. Caporale, and M. Bertogna, "Motion Planning and Control for Multi Vehicle Autonomous Racing at High Speeds," in *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, 2022, pp. 2775–2782.
- [2] M. Rowold, L. Ögretmen, T. Kerbl, and B. Lohmann, "Efficient spatiotemporal graph search for local trajectory planning on oval race tracks," *Actuators*, vol. 11, no. 11, p. 319, 2022.
- [3] L. Ögretmen, M. Rowold, M. Ochsenius, and B. Lohmann, "Smooth Trajectory Planning at the Handling Limits for Oval Racing," *Actuators*, vol. 11, no. 11, p. 318, 2022.
- [4] S. M. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.
- [5] J. H. Jeon, R. Cowlagi, S. Peters, S. Karaman, E. Frazzoli, P. Tsiotras, and K. Iagnemma, "Optimal motion planning with the half-car dynamical model for autonomous high-speed driving," in *2013 American Control Conference*, 2013, pp. 188–193.
- [6] J. Ziegler and C. Stiller, "Spatiotemporal state lattices for fast trajectory planning in dynamic on-road driving scenarios," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009, pp. 1879–1884.
- [7] K. Kant and S. Zucker, "Toward efficient trajectory planning: The path-velocity decomposition," *International Journal of Robotic Research - IJRR*, vol. 5, no. 3, pp. 72–89, 1986.
- [8] T. Stahl, A. Wischnewski, J. Betz, and M. Lienkamp, "Multilayer Graph-Based Trajectory Planning for Race Vehicles in Dynamic Scenarios," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, 2019, pp. 3149–3154.
- [9] A. Buyval, A. Gabdulin, R. Mustafin, and I. Shimchik, "Deriving overtaking strategy from nonlinear model predictive control for a race car," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 2623–2628.
- [10] D. Kalaria, P. Maheshwari, A. Jha, A. K. Issar, D. Chakravarty, S. Anwar, and A. Towar, "Local nmppc on global optimised path for autonomous racing," presented at the ICRA workshop on Opportunities and Challenges with Autonomous Racing, May 2021.
- [11] J. Bhargav, J. Betz, H. Zheng, and R. Mangharam, "Track based offline policy learning for overtaking maneuvers with autonomous racecars," presented at the ICRA Workshop on Opportunities and Challenges with Autonomous Racing, Jul. 2021.
- [12] S. He, J. Zeng, and K. Sreenath, "Autonomous racing with multiple vehicles using a parallelized optimization with safety guarantee using control barrier functions," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 3444–3451.
- [13] M. Rowold, L. Ögretmen, U. Kasolowsky, and B. Lohmann, "Online time - optimal trajectory planning on three-dimensional race tracks," in *2023 IEEE Intelligent Vehicles Symposium Proceedings*, 2023 (Submitted and Approved for Publication).
- [14] A. Liniger, A. Domahidi, and M. Morari, "Optimization-based autonomous racing of 1:43 scale RC cars," *Optimal Control Applications and Methods*, vol. 36, no. 5, pp. 628–647, 2015.
- [15] W. Lim, S. Lee, M. Sunwoo, and K. Jo, "Hierarchical trajectory planning of an autonomous car based on the integration of a sampling and an optimization method," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 2, pp. 613–626, 2018.
- [16] J. Ziegler, P. Bender, T. Dang, and C. Stiller, "Trajectory planning for bertha - a local, continuous method," in *2014 IEEE Intelligent Vehicles Symposium Proceedings*, 2014, pp. 450–457.
- [17] P. Bender, O. Tas, J. Ziegler, and C. Stiller, "The combinatorial aspect of motion planning: Maneuver variants in structured environments," in *2015 IEEE Intelligent Vehicles Symposium (IV)*, 2015, pp. 1386–1392.
- [18] G. Perantoni and D. J. N. Limebeer, "Optimal control of a formula one car on a three-dimensional track—part I: Track modeling and identification," *Journal of Dynamic Systems, Measurement, and Control*, vol. 137, no. 5, p. 051018, 2015.
- [19] T. Lozano-Pérez and M. Wesley, "An algorithm for planning collision-free paths among polyhedral obstacles," *Communications of the ACM*, vol. 22, no. 10, pp. 560–570, 1979.
- [20] U. Ramer, "An iterative procedure for the polygonal approximation of plane curves," *Computer Graphics and Image Processing*, vol. 1, no. 3, pp. 244–256, 1972.
- [21] E. Kerrigan and J. Maciejowski, "Soft constraints and exact penalty functions in model predictive control," in *Proc. UKACC International Conference (Control 2000)*, 2000, pp. 2319–2327.