2018 21st International Conference on Intelligent Transportation Systems (ITSC)
Maui, Hawaii, USA, November 4-7, 2018

Spatiotemporal Motion Planning with Combinatorial Reasoning for Autonomous Driving

Klemens Esterle[1], Patrick Hart[1], Julian Bernhard[1] and Alois Knoll[2]

***Abstract*— Motion planning for urban environments with numerous moving agents can be viewed as a combinatorial problem. With passing an obstacle before, after, right or left, there are multiple options an autonomous vehicle could choose to execute. These combinatorial aspects need to be taken into account in the planning framework. We address this problem by proposing a novel planning approach that combines trajectory planning and maneuver reasoning. We define a classification for dynamic obstacles along a reference curve that allows us to extract tactical decision sequences. We separate longitudinal and lateral movement to speed up the optimization-based trajectory planning. To map the set of obtained trajectories to maneuver variants, we define a semantic language to describe them. This allows us to choose an optimal trajectory while also ensuring maneuver consistency over time. We demonstrate the capabilities of our approach for a scenario that is still widely considered to be challenging.**

I. Introduction

A. Motivation

Autonomous driving intends to relieve the driver of the task of driving, thus promising great improvements in terms of safety and comfort. With encouraging solutions for the perception task enabled by deep learning, the behavior generation remains one of the biggest challenges for autonomous driving in order to achieve full autonomy. The behavior generation problem is to find an optimal motion regarding safety and comfort under the premise of obeying traffic rules, vehicle kinematics and dynamics. Satisfying real-time demands to ensure reactiveness to dynamic obstacles in critical scenarios is a key challenge for all motion planning algorithms [1].

A typical urban scene is presented in Fig. 1. The blue ego vehicle needs to overtake the stationary yellow vehicle and consider oncoming traffic and pedestrians crossing the street. Planning architectures that separate tactical maneuver selection and trajectory planning create handicaps in these types of situations. First of all, the separation may lead to sequences of high level actions that are physically not feasible. While this is typically handled by introducing additional safety margins, it limits the planner's ability to navigate in highly constrained environments with multiple obstacles. Second, if the tactical planner does not take the topology of the planning problem into account, the high-level sequence of actions passed to the trajectory planner may not be consistent with the past.

[1]Klemens Esterle, Patrick Hart and Julian Bernhard are with fortiss GmbH, An-Institut Technische Universität München, Munich, Germany
[2]Alois Knoll is with Robotics, Artificial Intelligence and Real-time Systems, Technische Universität München, Munich, Germany

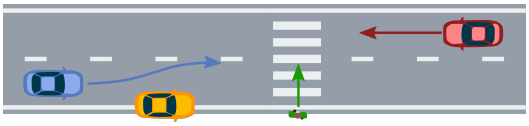

Fig. 1: A typical urban scenario: Pedestrians crossing the street, a parked vehicle (yellow) blocking part of the lane and oncoming traffic (red). The ego vehicle is displayed in blue.

B. Related Work

Spatiotemporal motion planning approaches can be divided into path-velocity decomposition approaches [2], sampling-based approaches [3, 4] and optimization methods using Model Predictive Control [5].

Planning architectures which decouple the spatiotemporal problem into path- and speed-profile planning reduce the computational costs by a considerable amount. The decomposition into a high-level path planning problem and a trajectory planning problem works well for traffic circles or simple crossings when a predefined path does not change. However, path-velocity decomposition provides poor results for complex scenarios with moving obstacles.

Sampling-based methods are able to deal with non-convex constraints. McNaughton *et al.* [3] present a spatial-temporal state-lattice-based approach based on dynamic programming. Due to the necessary state discretization with lattice-based methods, it is only suitable for highway driving. Werling *et al.* [4] propose a state-lattice based implementation that generates sampled quintic polynomials. The jerk-optimal trajectories are first selected in local coordinates and then transformed to Cartesian coordinates to check for collisions. This computationally expensive transformation combined with the curse of dimensionality resulting from a high state space discretization limits the ability of this approach to reactively avoid obstacles.

Creating multiple trajectories in a receding horizon fashion introduces the problem of consistent trajectory selection over time. Gu *et al.* [6] use a sampling-based trajectory planner to obtain trajectory candidates for the combinatorial motion planning problem. In order to avoid oscillation between multiple maneuvers, they group the generated trajectories by topological properties afterwards and impose consistency over time. Sontges and Althoff [7] use a similar concept of topological grouping for the analysis of reachable sets. Each driving corridor then corresponds to a different high-level decision sequence.

Local optimization methods formulating the motion problem as an optimal control problem do not suffer from any

978-1-7281-0323-5/18/$31.00 ©2018 IEEE

discretization errors in contrast to sampling based methods. Ziegler *et al.* [5] present a spatiotemporal non-linear local optimization scheme. Due to the non-linear model formulation, computation time highly depends on the quality of the initialization. As this approach only guarantees to find local optima, it requires a preprosessing layer decomposing the combinatorial space to set up collision constraints for each maneuver variant [8]. However, a generic constraint generation for complex scenarios still poses a major problem to the decomposition of the state space.

In order to deal with the combinatorial aspects, Zhan *et al.* [9] introduce a planning framework that plans longitudinal and lateral spatial movements separately to reduce computational costs. They use a layered graph-search approach and combine lateral and longitudinal motion using quadratic optimization. They classify environmental objects into point-overlap, line-overlap and undecided-overlap to deal with the combinatorial aspect. However, their search-based approach introduces longitudinally discretized actions that may not cover the optimal solution.

C. Contribution of this Paper

In this work, we propose a planning approach which plans trajectories for multiple maneuver types. We adapt the idea from [9] of a generic environmental representation of obstacles along the ego vehicle's reference curve but apply it to optimization based planning. This way, we acknowledge the combinatorial aspect of motion planning and reduce the number of maneuvers passed to the trajectory planner. We generate maneuver envelopes, that represent constraints to fully characterize a local trajectory optimization problem. Based on the optimization programs proposed in [10, 11], we separate longitudinal and lateral motion planning to reduce the computational costs. We calculate optimal trajectories in local coordinates for multiple maneuver types. We then need to select the best maneuver based on motion optimality and maneuver consistency, which is why we group the planned trajectories to maneuver variants. We will use the idea of topological trajectory grouping from [6] and apply it to an optimization-based trajectory planner to allow for a reasoning about the planned maneuver variant.

To summarize, we contribute

- a novel approach for a fused trajectory planning and maneuver selection, solving the existing problem of feasibility of pre-defined maneuvers,
- an optimization-based framework able to deal with combinatorial options and
- the demonstration of the technical abilities in a challenging scenario.

This work is further organized as follows: Section II defines the problem this paper aims to solve. The proposed method is presented in Section III. Section IV evaluates the algorithm's abilities followed by a discussion in Section V.

II. Problem Statement and Definitions

Given a reference curve as a series of points with no requirements on smoothness, we aim to find a collision-free and comfortable trajectory, roughly following the reference curve. The motion planner must account for static and dynamic obstacles. A superior strategy module responsible for high-level decisions like lane changes passes both a reference curve and a reference velocity v_{ref} to the trajectory planning module. This way, map-based information such as recommended velocities while turning can be incorporated in the planning module. The trajectory planner should output a sequence of states in world coordinates that can be passed to a trajectory tracking controller. This work omits the uncertainty about the state of other traffic participants.

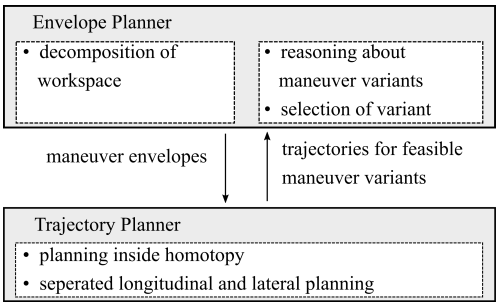

Fig. 2: Architectural overview of the proposed method consisting of an envelope planner passing homotopic planning candidates to a trajectory planner. The solved trajectories are passed back and evaluated for optimality and semantic consistency.

Fig. 2 shows an architectural overview of our approach. An envelope planner decomposes the spatiotemporal planning problem into multiple sub-problems, which we will call *maneuver envelopes*. These maneuver envelopes are then passed to the trajectory planner. Each envelope leads to a local optimization problem, for which a set of *homotopic trajectories* exists. Homotopic trajectories are co-terminal trajectories that can be continuously deformed from one to another without intersecting an obstacle [12]. We will use these maneuver envelopes to impose linear collision avoidance constraints to the local optimization problem instead of relying on a suitable initialization.

However, the maneuver envelopes do not contain the temporal passing order of the objects O_i, which motivates us to adapt the definition of a *maneuver variant* from [8] to be a set of homotopic trajectories. Following the ideas of [6], we distinguish different maneuver variants using topological distinction (*How does the trajectory avoid obstacles?*) and sequential distinction (*What overtaking order does it follow?*). Sontges and Althoff [7] argue that obeying the mathematical definition of homotopy does not lead to a grouping of trajectories suitable for autonomous driving. In order to semantically describe the planned trajectories for reasoning in III-C, we construct our semantic language L as following:

$$\mathrm{L} := \mathrm{L}\,\texttt{then}\,\mathrm{L} \parallel \mathcal{O}\,\texttt{is passed}\,\mathcal{H} \parallel \mathrm{L}\,\texttt{and}\,\mathrm{L} \quad (1)$$

with

$$\mathcal{O} := \{O_i\} \quad (2)$$

and

$$\mathcal{H} := \texttt{left} \parallel \texttt{right} \tag{3}$$

We state events to happen sequentially (`then`) or simultaneously (`and`).

III. Planning and Reasoning Framework

A. Combinatorial Decomposition

We introduce an environment representation and decompose the combination problem into convex sub-problems. We define longitudinal and lateral rules for formulating convex state constraints for each obstacle type. This set of rules essentially allows us to reduce the number of sub-problems.

Spatiotemporal Environmental Representation

Based on the predicted motion of the obstacle in relation to the reference curve of the ego vehicle, we construct convex hulls around the obstacle. We then derive free space-time decision envelopes $\zeta_{t_p}^{o,\Delta}$ for each obstacle o and decision Δ in a local reference system as

$$\zeta_{t_p}^{o,\Delta} = \begin{bmatrix} \zeta_{long,max}^{o,\Delta} \\ \zeta_{long,min}^{o,\Delta} \\ \zeta_{lat,max}^{o,\Delta} \\ \zeta_{lat,min}^{o,\Delta} \end{bmatrix} = \begin{bmatrix} s_{max} \\ s_{min} \\ d_{left}(s_i) \\ d_{right}(s_i) \end{bmatrix} \in R^{2+2n_s} \tag{4}$$

at each prediction time step t_p with the arc length s and the perpendicular offset d. n_s denotes the number of spatial sampling points s_i. This formulation allows us to represent obstacle-related longitudinal and lateral constraints in local coordinates independently of the future ego motion of the vehicle.

The longitudinal constraints are derived by calculating the spatial length at which the projected occupied space of the obstacle overlaps with the free-space of the ego vehicle along the reference line. We approximate this free-space with a polygon with the width of the ego vehicle's current lane. The lateral constrains are derived based on a distance calculation between the reference line and the obstacle at each spatial support point s_i.

Inspired by Zhan *et al.* [9], we classify obstacles into *non-overlapping* (Fig. 3(a)), *line-overlapping* (Fig. 4(a)) and *point-overlapping* (Fig. 5(a)) obstacles. For *non-overlapping* obstacles, there exists no intersection between the obstacle's predicted future motion and the reference path of the ego vehicle. Fig. 3(a) shows an example where the obstacle is parallel to the reference path (e.g. oncoming traffic). We limit the set of tactical decisions for this class of obstacles by stating that the ego vehicle should only pass the obstacle on the side of the reference path (Table I). Therefore, a decision on which side to avoid the obstacle is not needed. Fig. 3(b) displays the resulting state constraints and a possible ego trajectory avoiding the obstacle.

Line-overlapping obstacles are characterized through a line-wise overlapping to the vehicle's configuration space

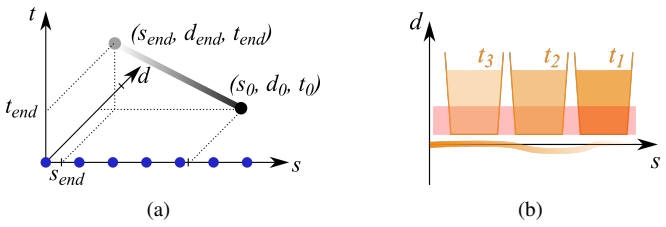

Fig. 3: (a) illustrates the motion of a non-overlapping obstacle in a local reference frame along the reference curve with the arc length s and the lateral offset d. Blue reference points represent the ego vehicle's reference curve. As there is no overlap with the reference curve, the obstacle only needs to be considered laterally. (b) shows the resulting lateral constraints (orange). The projected occupied space of the obstacle is shown in red. A possible trajectory of the ego vehicle is color-coded in orange indicating temporal progress.

along the reference curve. This could for example be a preceding vehicle. Fig. 4(b) shows the imposed state constraints by a preceding vehicle slowing down the ego vehicle.

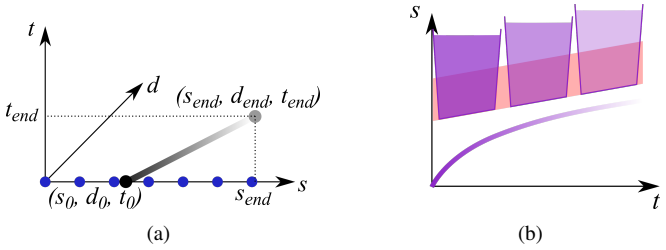

Fig. 4: (a) shows the motion of a line-overlapping obstacle. As displayed in (b), the obstacle type needs to be accounted for longitudinally, imposing state constraints on the spatial coordinate s.

Point-overlapping obstacles have a fixed entry point and a fixed exit point from the configuration-space defined through the reference path. This could be vehicles at an intersection, pedestrians on a crosswalk or other agents intersecting with the reference path. Fig. 5 displays our constraint formulations for the combinatorial options for a *point-overlapping* obstacle. We define four possible options: Passing the pedestrian before or after, or avoiding it on the left or right, see Fig. 5(c)). For each of them, we derive longitudinal and lateral state constraints. As Fig. 5(e) shows, when maintaining speed intending to avoid the obstacle, lateral constraints for left or right need to be considered. The lateral constraints for passing before and after are illustrated in Fig. 5(d) and 5(f).

The possible options for each obstacle class are summarized in Table I.

TABLE I: Tactical decisions for passing or avoiding obstacles.

Obstacle Type	Possible Tactical Decisions Δ_j			
	Δ_1	Δ_2	Δ_3	Δ_4
Point-overlapping	before	after	right	left
Line-overlapping		after	right	left
Non-overlapping			right/left	

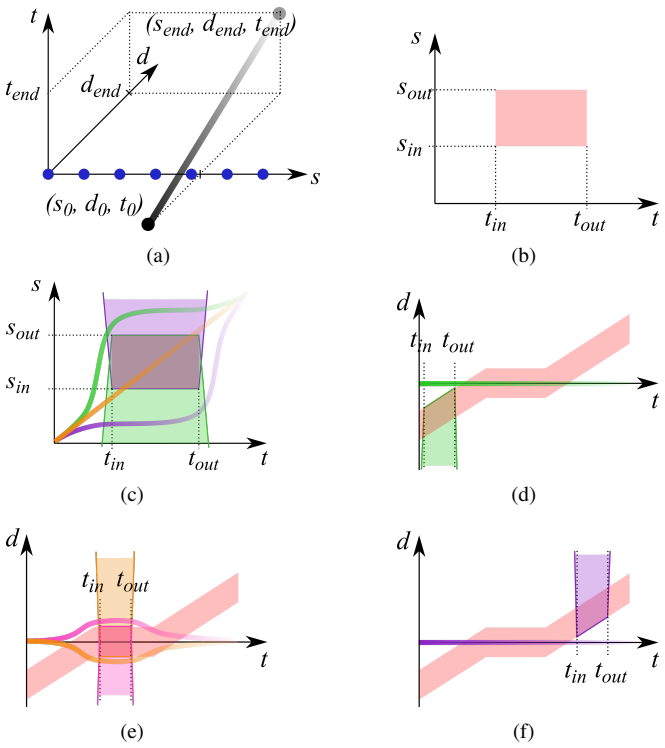

Fig. 5: (a) illustrates the motion of a point-overlapping obstacle. (c) shows the combinatorial options passing the obstacle before (green), after (purple) or avoiding it left or right (orange). The respective lateral constraints for the maneuvers passing before (d), avoiding left or right (magenta and orange) in (e)) and passing after (f) are displayed.

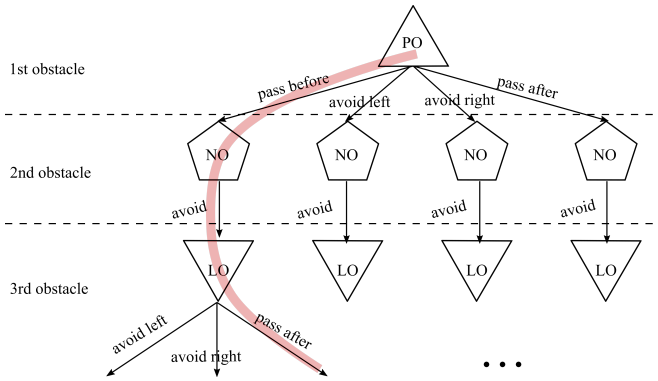

Fig. 6: Tactical decisions for a scene with three obstacles: a point-overlapping obstacle (1st), a non-overlapping obstacle (2nd) and a line-overlapping obstacle (3rd). The red line indicates a tactical decision sequence, that represents a maneuver envelope. The sequential ordering of the objects is not encoded here.

Combinatorial Planning Scheme

For each prediction time step t_p, we derive the maneuver envelopes $\zeta_{t_p}^m$ by creating all combinations of the obstacle's free space-time decisions envelopes $\zeta_{t_p}^{\Delta}$ and merging them according to

$$\zeta_{t_p}^m = \begin{cases} \min & \zeta_{long,max}^{o,\Delta} \\ \max & \zeta_{long,min}^{o,\Delta} \\ \min & \zeta_{lat,max}^{o,\Delta} \\ \max & \zeta_{lat,min}^{o,\Delta} \end{cases} \quad \forall o_i, \Delta_j. \tag{5}$$

Fig. 6 displays a tree with the tactical decisions for a set of a point-overlapping, a non-overlapping and a line-overlapping obstacle. Traversing the tree from the root to a leaf will be called *maneuver sequence* m for the remainder of this paper. Invalid maneuver envelopes are pruned, such as if $d_{left} < d_{right}$ with $d_{left} > 0$ and $d_{right} > 0$. As the number of sequences grows exponentially with the number of obstacles, we plan to reduce this set even further in the future by the use of heuristics.

The valid maneuver envelopes are passed to the trajectory planner. The constraints of each maneuver envelope form a convex state space, making sure the trajectory planner can only converge to one optimum. First, a longitudinal trajectory candidate is generated for each envelope. If the problem is infeasible, the formulation as a linear quadratic program allows us to quickly terminate the optimization if it does not converge. For all longitudinal candidates, we optimize the lateral behavior. If successful, we add the trajectory to the set of possible maneuvers. By reason of functional safety, we add a collision check in Cartesian coordinates for the projected motion to make sure that the coordinate transformations did not introduce any error that may lead to collisions. The optimal trajectory in regard to a set of criteria can then be selected from the set of possible trajectories (see Section III-C). The model predictive control is sped up by reusing the previous solution as an initialization.

B. Trajectory Planning

The trajectory planning algorithm is running in a receding horizon fashion. In this section, we will mainly discuss a single optimization run representing a Model Predictive Control stage. In order to separate longitudinal and lateral motion, the trajectory optimization problem needs to be defined in a local reference frame as in [10, 13], so-called *Frenet coordinates*. Frenet coordinates define the motion along a reference curve $\Gamma(s)$ through the arc length s and the lateral offset d (see Fig. 7).

The decomposition into longitudinal and lateral motion allows us to handle the desired behavior separately, as e.g. the longitudinal motion may be subject to a high-level decision making entity. The spatial decomposition also simplifies the computational complexity of the motion planning scheme as it allows to formulate the trajectory planning problem through two linear sequential optimization programs. Additionally, the separated motion formulation allows to construct linear safety constraints for each motion.

In the following paragraph, we present the key ideas from the optimization schemes introduced in [10, 11]. As the formulation uses a linearized model, no iterative scheme for approximating the non-linearity is needed, which significantly simplifies the problem and reduces the computational complexity.

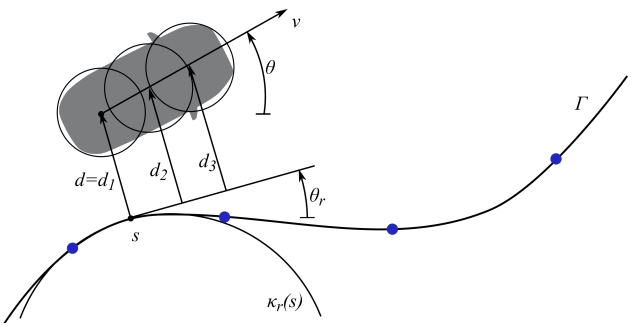

Fig. 7: Vehicle model along a reference curve Γ in local street coordinates [10]. The blue waypoints indicate a discrete representation of the reference curve. The arc length s, the perpendicular offset d to the reference curve and the orientation of the vehicle θ define the vehicle state. d_1, d_2, d_3 define the system output.

Optimal Control Problem

Both longitudinal and lateral optimization schemes introduced in [10, 11] formulate constrained optimal control problems. With a linear model and a quadratic cost function, the control problem can be formulated over a prediction span N and solved using the batch method [14]. Given a state matrix $\boldsymbol{A}(t)$, an input matrix $\boldsymbol{B}(t)$, an error matrix $\boldsymbol{E}(t)$ and an output matrix $\boldsymbol{C}(t)$, the continuous system model

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}(t)\boldsymbol{x}(t) + \boldsymbol{B}(t)\boldsymbol{u}(t) + \boldsymbol{E}(t)\boldsymbol{z}(t) \tag{6}$$

$$\boldsymbol{y}(t) = \boldsymbol{C}(t)\boldsymbol{x}(t) \tag{7}$$

can be discretized using Euler integration leading to

$$\hat{\boldsymbol{x}}(k+1) = \hat{\boldsymbol{A}}(k)\boldsymbol{x}(k) + \hat{\boldsymbol{B}}(k)\hat{\boldsymbol{u}}(k) + \hat{\boldsymbol{E}}(k)\hat{\boldsymbol{z}}(k) \tag{8}$$

$$\hat{\boldsymbol{y}}(k) = \hat{\boldsymbol{C}}(k)\hat{\boldsymbol{x}}(k). \tag{9}$$

By stacking together sequential states using the batch method, the quadratic program is transformed into a sequential quadratic program which can be solved using standard SQP solvers. With the discrete step size k being represented by an index, the state vector sequence can be written as

$$\mathbf{x} = \left[\hat{\boldsymbol{x}}_1^T, \quad \ldots, \quad \hat{\boldsymbol{x}}_N^T\right]. \tag{10}$$

The state input $\mathbf{u}$, the state output $\mathbf{y}$ and the error $\mathbf{z}$ are defined analogously. The costs can then be formulated as

$$J(\mathbf{x}, \mathbf{u}, \mathbf{x}_{ref}) = [\mathbf{x} - \mathbf{x}_{ref}]^T \mathcal{Q} [\mathbf{x} - \mathbf{x}_{ref}] + \mathbf{u}^T \mathcal{R} \mathbf{u}, \tag{11}$$

where $\mathcal{R}$ and $\mathcal{Q}$ denote cost weight matrices and $\mathbf{x}_{ref}$ denotes the reference state. Formulating states x_j as outputs y_j allows to express systems constraints as input constraints $u_{c,j}$.

Longitudinal Trajectory Planning

We use the linear time-variant system model presented by Gutjahr *et al.* [11]. The system model simplifies to

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}(t)\boldsymbol{x}(t) + \boldsymbol{B}(t)\boldsymbol{u}(t), \tag{12}$$

$$\boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{x}(t), \tag{13}$$

where $\boldsymbol{x} = \left[s, v, a, j\right]$ denotes the state and $\boldsymbol{u} = \left[\ddot{a}\right]$ denotes the input. Gutjahr *et al.* [11] define the system output to constrain s and a. We extend this to include the velocity v, as it allows us to prohibit backward motion. The system output is thus defined as $\boldsymbol{y} = \left[s, v, a\right]$. The respective continuous state matrices can be found in the Appendix. For the cost function (11), the cost weight matrices $\hat{\boldsymbol{Q}}(k)$ and $\hat{\boldsymbol{R}}(k)$ are defined as

$$\hat{\boldsymbol{Q}}(k) = \mathrm{diag}(w_s(k), w_v(k), w_a(k), w_j(k)) \tag{14}$$

and

$$\hat{\boldsymbol{R}}(k) = \left[w_u(k)\right]. \tag{15}$$

We end up solving the following longitudinal planning problem for each valid maneuver envelope m:

$$\begin{aligned}
\forall m \quad \min_{u} \quad & J(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{x}_{ref}) && \text{(16a)}\\
s.t. \quad & \forall t \in [t_0, t_0 + T] && \text{(16b)}\\
& \dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), \boldsymbol{u}(t)) && \text{(16c)}\\
& \boldsymbol{u}_{min} \leq \boldsymbol{u}(t) \leq \boldsymbol{u}_{max} && \text{(16d)}\\
& \boldsymbol{y}^m_{min}(t) \leq \boldsymbol{y}(t) \leq \boldsymbol{y}^m_{max}(t) && \text{(16e)}
\end{aligned}$$

with

$$\boldsymbol{y}^m_{max}(t) = \left[\zeta^m_{long,max}(t), \quad v_{max}(t), \quad a_{max}(t)\right], \tag{17a}$$

$$\boldsymbol{y}^m_{min}(t) = \left[\zeta^m_{long,min}(t), \quad v_{min}(t), \quad a_{min}(t)\right]. \tag{17b}$$

We express the output constraints as input constraints using batch matrices, cf. [11].

Lateral Trajectory Planning

We use the linear time-variant system model presented in [10] with the state $\boldsymbol{x} = \left[d, \theta, \kappa, \theta_r, \kappa_r\right]$, the input $\boldsymbol{u} = \left[\dot{\kappa}\right]$, the output $\boldsymbol{y} = \left[d_1, d_2, d_3, \kappa\right]$ and the error $\boldsymbol{z} = \left[\kappa_r\right]$. The respective continuous state matrices can be found in the Appendix. The cost function minimizes the distance to the reference line d, the orientation difference between the vehicle and the reference line $\theta - \theta_r$, the curvature κ and the change of curvature $\dot{\kappa}$. The usage of an explicit reference state $\boldsymbol{x}_{ref}$ is not necessary, as all state variables are desired to be zero. Formally, this leads to

$$\boldsymbol{Q}(k) = \begin{bmatrix} w_d(k) & 0 & 0 & 0 & 0 \\ 0 & w_\theta(k) & 0 & -w_\theta(k) & 0 \\ 0 & 0 & w_\kappa(k) & 0 & 0 \\ 0 & -w_\theta(k) & 0 & w_\theta(k) & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{18}$$

and

$$\boldsymbol{R}(k) = \left[w_u(k)\right]. \tag{19}$$

The linearized model is only valid for small deviations $\theta - \theta_r$. The formulation can deal with reference curves even with high and discrete curvatures. This would allow the usage of a search-based path planner without a computationally costly post-optimization such as the one from [15]. The distance to obstacles is calculated based on the static reference curve. This means that the otherwise costly collision check is computed only once for each successful optimization instead

of for each iteration step. Finally, we solve the following lateral optimization problem for all valid maneuvers m, for which the longitudinal planning succeeded.

$$\begin{aligned} \forall m \quad \min_{u} \quad & J(\boldsymbol{x}, \boldsymbol{u}) && \text{(20a)} \\ s.t. \quad & \forall t \in [t_0, t_0 + T] && \text{(20b)} \\ & \dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), \boldsymbol{u}(t)) && \text{(20c)} \\ & \boldsymbol{u}_{min} \leq \boldsymbol{u}(t) \leq \boldsymbol{u}_{max} && \text{(20d)} \\ & \boldsymbol{y}^m_{min}(t) \leq \boldsymbol{y}(t) \leq \boldsymbol{y}^m_{max}(t) && \text{(20e)} \end{aligned}$$

with

$$\boldsymbol{y}^m_{max} = \left[\tilde{\boldsymbol{\zeta}}^m_{lat,max}(t), \tilde{\boldsymbol{\zeta}}^m_{lat,max}(t), \tilde{\boldsymbol{\zeta}}^m_{lat,max}(t), \kappa_{max}(t)\right], \tag{21a}$$

$$\boldsymbol{y}^m_{min} = \left[\tilde{\boldsymbol{\zeta}}^m_{lat,min}(t), \tilde{\boldsymbol{\zeta}}^m_{lat,min}(t), \tilde{\boldsymbol{\zeta}}^m_{lat,min}(t), \kappa_{min}(t)\right]. \tag{21b}$$

We make use of the batch approach again to express output as input constraints, cf. [10]. From the longitudinal planning, we know the longitudinal motion $s(t)$, which we use to transform the spatial-dependent $\boldsymbol{\zeta}^m_{lat}$ to time-dependent $\tilde{\boldsymbol{\zeta}}^m_{lat}$. Similar to [10], we use slack variables to relax the constraints for $d_{1,2,3}$ for the first three optimization support points, as the model error leads to infeasibilities close to obstacles when being used in a receding horizon fashion.

C. Reasoning and Maneuver Variant Selection

After the construction of homotopic maneuver envelopes and their trajectory optimization, we need to select the best trajectory. For this, we first define our trajectory costs as

$$J_{traj} = \frac{1}{n}\sum_{i=1}^{n}\left[w_{r,a}a_i^2 + w_{r,j}j_i^2 + w_{r,d}\big(d_i - d_{d,i}\big)^2 + w_{r,\dot{\kappa}}\dot{\kappa_i}^2\right]. \tag{22}$$

We incorporate comfort measures with the curvature derivative $\dot{\kappa_i}$, the longitudinal acceleration a_i and the jerk j_i. We also integrate the proximity to other obstacles $d_i - d_{ref,i}$ as a safety measure into the cost functional. $d_{ref,i}$ represents the maximum possible distance to the left and right side

$$d_{ref,i} = \frac{1}{2}\Big(d_{left,i} + d_{right,i}\Big). \tag{23}$$

Despite using an optimization-based planning approach, the temporal consistency of the selected maneuver will not necessarily hold, which potentially may lead to oscillating behavior. First of all, the receding horizon concept and the uncertainty of the environment provide new information to the trajectory planner at every planning stage. Second, the solutions obtained from the trajectory planner are only suboptimal as we simplify the trajectory planning to two separate planning tasks. This motivates to include a consistency cost term J_{cons} which penalizes switching between maneuver sequences. To quantify the costs, we need to describe the semantic consistency of the maneuver to the one previously selected. The semantic description needs to contain both topological as well as sequential information. As the maneuver envelopes from Section III-A do not contain sequential information, we refer to the semantic language which we defined in Section II to map our obtained trajectories back to a maneuver variant. Similarly to [7], we construct two-dimensional surfaces U_i along s and t in the three-dimensional domain for each obstacle O_i. The intersection points between the ego trajectory τ and U_i yield the sequential information t_i. The signed distance to the obstacle at t_i yields the topological information. This can be seen as an abstract function mapping possibly many trajectories onto one maneuver variant.

TABLE II: Weights for trajectory optimization

w_s	w_v	w_a	w_j	$w_{\dot{j}}$	w_d	w_θ	w_κ	$w_{\dot{\kappa}}$
0	10^3	10	10^2	10^3	10^2	10	10	10^3

TABLE III: Weights for reasoning

$w_{r,a}$	$w_{r,j}$	$w_{r,d}$	$w_{r,\dot{\kappa}}$	$w_{r,c}$
10	10	10^2	10^3	30

With these information, the semantic description for a maneuver variant can be automatically extracted from a given trajectory. It allows us to reason about the related maneuver variants and calculate the consistency costs

$$J_{cons} = w_{r,c}\Big(N_{max} - \sum_{j=1}^{M}\delta(L_{j,t_i} \neq L_{j,t_{i-1}})\Big), \tag{24}$$

where N_{max} denotes the maximum number of elements in the language sequences at time t_i, M the number of similar language items between the description $L_{t_i,j}$ at the current time step and the previous time step $L_{t_{i-1},j}$. For each similar item L_j in shared order, $\delta(\cdot)$ outputs 1, otherwise 0. This semantic reasoning adds an additional safety layer to our selection process. In the future, it could be enhanced by adding uncertainty information from the object detection module or incorporating occlusion.

The total costs J_{total} for the selection process are then defined by

$$J_{total} = J_{traj} + J_{cons}\,. \tag{25}$$

IV. Evaluation

We apply the proposed method to an urban driving scenario similar to the motivating example of this paper (see Fig. 1) with a crossing pedestrian O_1 at $v = 0.5\mathrm{m\,s^{-1}}$ (green), a static obstacle O_2 (yellow) and oncoming traffic O_3 at $v = 10\mathrm{m\,s^{-1}}$ (red). The chosen parameters are displayed in Table II, Table III and Table IV.

Fig. 8 shows the free space-time envelopes $\boldsymbol{\zeta}^{o,\Delta}_{t_p}$ at a single prediction step $t_p = 2.4\mathrm{s}$ for the maneuver envelope m_{15}. With the decision to pass the pedestrian on the right, the

TABLE IV: Parameters for planning and prediction horizons

Horizon	N	dt[s]
Longitudinal Trajectory Optimization	20	0.2s
Lateral Trajectory Optimization	20	0.2s
Prediction	10	0.4s

TABLE V: Maneuver envelopes m_{14} and m_{15} with decisions Δ_j for each obstacle O_i.

Maneuver Envelope m	Decisions Δ_j		
	O_1	O_2	O_3
m_{14}	after	right	left
m_{15}	right	right	left

free-space resembles that by the left boundary going around the predicted occupancy of the pedestrian. The predicted oncoming traffic participant does not intersect with the lane of the ego vehicle and thus with the free-space. Based on the decision to pass the standing vehicle on the left, the right boundary goes of the free-space goes around the obstacle. The merged free space-time maneuver envelope $\zeta_{t_p}^{m}$ is displayed below (blue). In cartesian coordinates, it resembles the intersection of all the decision envelopes $\zeta_{t_p}^{o_{1-3},\Delta}$. As we operate in Frenet coordinates, we calculate it according to equation (5).

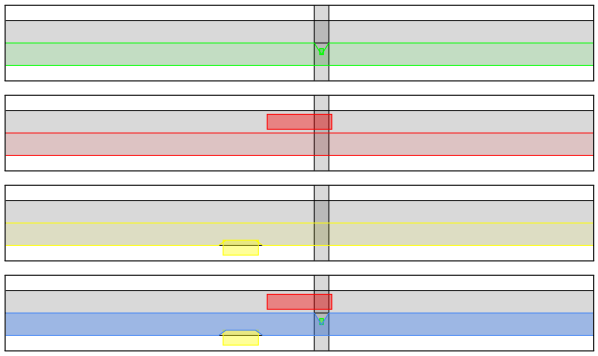

Fig. 8: Free space-time decision envelopes $\zeta_{t_{p6}|t_0}^{o,\Delta}$ for each obstacle at $t_0 = 0$s for the prediction step $t_{p6} = 2.4$s for the maneuver envelope m_{15}. The merged free space-time maneuver envelope $\zeta_{t_{p6}|t_0}^{m_{15}}$ is displayed below.

Fig. 9 shows the outcome of the full simulation over time. At $t_0 = 0$s, the planner chooses to execute m_{14}, which means passing after the pedestrian. See Table V for a full explanation of the maneuver. Note that for t_0, $\zeta_{t_{p1}|t_0}^{m_{14}}$ includes the full lane length, as the pedestrian will not have entered the lane at t_{p1}. For all following prediction steps, the pedestrian has to be taken into account either through longitudinal or lateral decision making. Because of this, the free-space envelopes $\zeta_{t_{p2\dots N}|t_0}^{m_{14}}$ end before the pedestrian.

At $t_6 = 1.2$s, the planner changes from m_{14} to m_{15}. $\zeta_{t_{p1\dots N}|t_6}^{m_{15}}$ thus do not stop before the pedestrian, but avoid him on the right, similar to Fig. 8.

The ego vehicle thus avoids the stationary vehicle, goes back to the ego lane and around the pedestrian while maintaining its initial speed. By modifying the optimality criteria, we could select a more defensive driving style.

V. Conclusion and Future Work

In this work, we proposed a fused trajectory optimization and maneuver selection by introducing a generic decision mechanism to derive maneuver sequences and a semantic

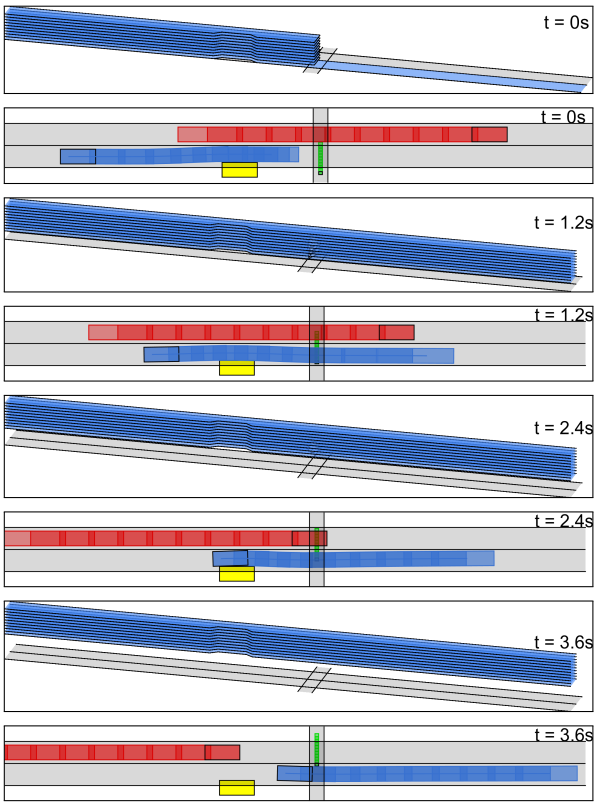

Fig. 9: Vehicle movement at $t_0 = 0$s, $t_6 = 1.2$s , $t_{12} = 2.4$s and $t_{18} = 3.6$s. The optimal free space-time maneuver envelopes $\zeta_{t_p}^{m}$ are also displayed for the each prediction time step.

language to reason about the maneuver of each obtained trajectory. By separating longitudinal and lateral motion in the trajectory planner, we simplify the constraint formulation as well as the planning problem, which thus allows us to compute multiple trajectories. Note that the maneuver selection framework could be used with other trajectory planners as well. As demonstrated in the simulation results, the novel approach can plan comfortable and safe spatiotemporal trajectories in complex urban driving scenarios.

The growing number of maneuver types with the number of obstacles still poses a major problem. We will investigate other approaches for the spatiotemporal topological analysis in the future, with the emphasis of discarding infeasible maneuver types. Machine Learning could be used as a heuristic to reduce the number of combinatorial sub-problems. Mixed integer quadratic programming could be investigated to improve the selection of the best trajectory. Other semantic information such as traffic rules could be incorporated into the semantic selection process. Accelerating the computation of the decision envelopes will need to be addressed in future work. We plan to implement the framework in C++ to investigate and improve the real-time capabilities of our approach and to allow for a real-world validation on a full-size research vehicle. In the future, we plan to investigate the possibility of incorporating prediction uncertainty and interaction awareness and the robustness against occlusions.

REFERENCES

[1] D. González, J. Pérez, V. Milanés, and F. Nashashibi, "A Review of Motion Planning Techniques for Automated Vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1135–1145, 2016.

[2] X. Qian, I. Navarro, A. de La Fortelle, and F. Moutarde, "Motion Planning for urban autonomous driving using Bézier Curves and MPC," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, 2016, pp. 826–833.

[3] M. McNaughton, C. Urmson, J. M. Dolan, and J.-W. Lee, "Motion planning for autonomous driving with a conformal spatiotemporal lattice," in *2011 IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 4889–4895.

[4] M. Werling, J. Ziegler, S. Kammel, and S. Thrun, "Optimal trajectory generation for dynamic street scenarios in a Frenét Frame," in *2010 IEEE International Conference on Robotics and Automation (ICRA)*, 2010, pp. 987–993.

[5] J. Ziegler, P. Bender, T. Dang, and C. Stiller, "Trajectory planning for Bertha – A local, continuous method," in *IEEE Intelligent Vehicles Symposium (IV)*, 2014, pp. 450–457.

[6] T. Gu, J. M. Dolan, and J.-W. Lee, "Automated Tactical Maneuver Discovery, Reasoning and Trajectory Planning for Autonomous Driving," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2016, pp. 5474–5480.

[7] S. Sontges and M. Althoff, "Computing possible driving corridors for automated vehicles," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, 2017, pp. 160–166.

[8] P. Bender, Ö. S. 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.

[9] W. Zhan, J. Chen, C.-Y. Chan, C. Liu, and M. Tomizuka, "Spatially-partitioned environmental representation and planning architecture for on-road autonomous driving," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2017, pp. 632–639.

[10] 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, pp. 1–10, 2016.

[11] B. Gutjahr, C. Pek, L. Gröll, and M. Werling, "Recheneffiziente Trajektorienoptimierung für Fahrzeuge mittels quadratischem Programm," *At-Automatisierungstechnik*, vol. 64, no. 10, pp. 786–794, 2016.

[12] S. Kim, K. Sreenath, S. Bhattacharya, and V. Kumar, "Trajectory Planning for Systems with Homotopy Class Constraints," in *Latest Advances in Robot Kinematics*, Dordrecht: Springer Netherlands, 2012, pp. 83–90.

[13] Y. Gao, A. Gray, J. V. Frasch, *et al.*, "Spatial Predictive Control for Agile Semi-Autonomous Ground Vehicles," in *11th International Symposium on Advanced Vehicle Control*, 2012.

[14] F. Borrelli, A. Bemporad, and M. Morari, *Predictive control for linear and hybrid systems*. Cambridge University Press, 2017.

[15] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Path planning for autonomous vehicles in unknown semi-structured environments," *International Journal of Robotics Research*, vol. 29, no. 5, pp. 485–501, 2010.

APPENDIX

A. Notes on Longitudinal Optimal Control Problem

The continuous system matrices for the longitudinal planning scheme stated in Section III-B are

$$\boldsymbol{A}(t) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \tag{26}$$

$$\boldsymbol{B}(t) = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix} \tag{27}$$

and

$$\boldsymbol{C}(t) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{28}$$

B. Notes on Lateral Optimal Control Problem

For the lateral planning problem described in Section III-B, the system matrices are

$$\boldsymbol{A}(t) = \begin{bmatrix} 0 & v(t) & 0 & -v(t) & 0 \\ 0 & 0 & v(t) & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & v(t) \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \tag{29}$$

$$\boldsymbol{B}(t) = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \end{bmatrix}, \tag{30}$$

$$\boldsymbol{C}(t) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & l/2 & 0 & -l/2 & 0 \\ 1 & l & 0 & -l & 0 \\ 0 & 0 & 1 & 0 & v(t) \end{bmatrix} \tag{31}$$

and

$$\boldsymbol{E}(t) = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{32}$$

C. Notes on Batch Formulation for Optimal Control Problem

The batch matrices are defined as following:

$$\mathcal{A} = \begin{bmatrix} \boldsymbol{A}_0^T & \left(\prod_{q=0}^{1} \boldsymbol{A}_{1-q}\right)^T & \dots & \left(\prod_{q=0}^{N-1} \boldsymbol{A}_{N-1-q}\right)^T \end{bmatrix} \tag{33}$$

$$\mathcal{B} = \begin{bmatrix} \boldsymbol{B}_0 & 0 & \dots & 0 \\ \boldsymbol{A}_1\boldsymbol{B}_0 & \boldsymbol{B}_1 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \left(\prod_{q=1}^{N-1} \boldsymbol{A}_{N+0-q}\right)\boldsymbol{B}_0 & \dots & \boldsymbol{A}_{N-1}\boldsymbol{B}_{N-2} & \boldsymbol{B}_{N-1} \end{bmatrix} \tag{34}$$

$$\mathcal{C} = \begin{bmatrix} \boldsymbol{C}_1 & & \\ & \ddots & \\ & & \boldsymbol{C}_N \end{bmatrix} \tag{35}$$

$\mathcal{E}$ is derived similarly to $\mathcal{B}$.