Curvature Aware Motion Planning with Closed-Loop Rapidly-exploring Random Trees

Berend van den Berg[1,2], Bruno Brito[1], Mohsen Alirezaei[3,4] and Javier Alonso-Mora[1]

***Abstract*— The road's geometry strongly influences the path planner's performance, critical for autonomous navigation in high-speed dynamic scenarios (e.g., highways). Hence, this paper introduces the Curvature-aware Rapidly-exploring Random Trees (CA-CL-RRT), whose planning performance is invariant to the road's geometry. We propose a transformation strategy that allows us to plan on a virtual straightened road and then convert the planned motion to the curved road. It is shown that the proposed approach substantially improves path planning performance on curved roads as compared to prior RRT-based path planners. Moreover, the proposed CA-CL-RRT is combined with a Local Model Predictive Contour Controller (LMPCC) for path tracking while ensuring collision avoidance through constraint satisfaction. We present quantitative and qualitative performance results in two navigation scenarios: dynamic collision avoidance and structured highway driving. The results demonstrate that our proposed navigation framework improves the path quality on curved highway roads and collision avoidance with dynamic obstacles.**

I. Introduction

Autonomous Driving requires a motion planner to find a motion plan from its current position to a goal position. A motion planner suitable for Autonomous Driving must be able to plan in structured (e.g., highways) and unstructured environments (e.g., parking lot). Typically, in unstructured environments, there are low speed limits and vehicle maneuverability is most important for the planner. In contrast, in structured environments like highway, the road geometry is simple and high speed limits make the dynamic feasibility of the motion plan crucial for safe driving.

Many studies implement motion planners that can plan in either structured [1], [2], [3], or unstructured [4], [5] environments. Some even solve a wider range of environments [6], [7], [8]. Yet, the generic implementation of a single motion planning method that can plan in the full range of environments is still an open question. Rapidly-exploring Random Trees have proven to be effective for planning in unstructured environments. However, implementing these methods real-time while ensuring a natural driving style on structured roads, remains to be solved.

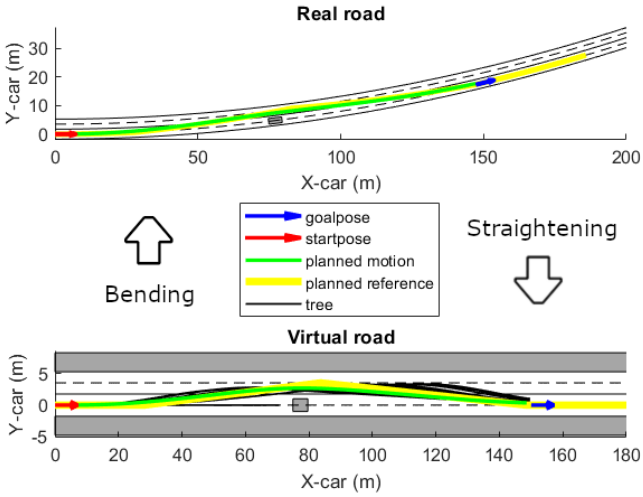


Fig. 1: The proposed motion planning method applied during high-speed driving on a curved highway.

In this paper, we address the identified gap in literature in the following way. Firstly, we introduce the Curvature-Aware Closed-Loop Rapidly-exploring Random Trees (CA-CL-RRT), whose planning performance is invariant to the road's geometry. A transformation strategy is proposed that allows to plan on a virtual straightened road and then convert the planned motion to the curved road (see Fig. 1). Secondly, we expand our previous work on LMPCC [9] to track a continuously updated path provided by the CA-CL-RRT. The proposed motion planning architecture allows using the same algorithm for maneuvering in unstructured environments, high-speed autonomous driving on structured roads (e.g., highways), and dynamic collision avoidance.

This paper is organized as follows: first we discuss state-of-the-art methods based on Rapidly-exploring Random Trees (RRTs) and present the contributions. Next, the motion planning problem and the preliminaries needed for solving the problem are introduced. Then, we present an expanded formulation of the Closed Loop Rapidly-exploring Random Tree (CL-RRT) method for curved roads and present the proposed motion planning framework. Lastly, we present simulation results for two distinct scenarios: autonomous navigation in highways with curved roads, and dynamic collision avoidance of two pedestrians crossing a road.

This work was supported by the Amsterdam Institute for Advanced Metropolitan Solutions, the Netherlands Organisation for Scientific Research (NWO) domain Applied Sciences (Veni 15916).

[1]The authors are with the Department of Cognitive Robotics, Delft University of Technology, 2628 CD Delft, The Netherlands {bruno.debrito, j.alonsomora}@tudelft.nl

[2]The author is with the Department of Mathematics and Computer Science, Eindhoven University of Technology, 5600 MB, Eindhoven, The Netherlands b.h.a.v.d.berg@tue.nl

[3]The author is with the department of Mechanical Engineering, Eindhoven University of Technology, 5600 MB, Eindhoven, The Netherlands m.alirezaei@tue.nl

[4]The author is with Siemens Industry Software and Services B.V., Digital Industry – Software – Simulation and Testing Services, 5708 JZ, Helmond, The Netherlands mohsen.alirezaei@siemens.com

A. Related work

RRTs [10] can quickly search in high-dimensional spaces that have complex constraints, which makes them extremely useful for real-time applications. The algorithm incrementally builds a tree of paths by connecting randomly sampled configurations using an exact [11] or approximate [10] steering function, based on vehicle kinematics or dynamics. The advantage of approximate steering methods is that they avoid the need for solving a Boundary Value Problem (BVP) for each sample, which can be a computationally expensive operation for differentially constrained systems [12].

One of the limitations of RRT-based methods is that they often converge to sub-optimal solutions [11], which expresses itself in path quality by e.g., a meandering path. Early variants address this issue by adding an optimization heuristic during nearest neighbor selection [7], [13], [14], or prune the path after a solution is found [15]. However, this does not guarantee asymptotic optimality. Therefore, the RRT* introduces a rewiring operation that can guarantee asymptotic convergence towards the optimal solution [11]. However, this method is currently restricted to using exact steering functions only, because of the number of BVP to solve during rewiring (e.g., [16]).

To avoid the need for solving a BVP, [7], [17] introduced the CL-RRT. This variant samples in the controller's input space (e.g., a path and reference velocity) instead of the vehicle inputs (e.g., steering angle and velocity). The controller inputs are used for simulating a closed-loop trajectory of the vehicle controlled by a lateral and a longitudinal controller.

Traditional RRT algorithms sample the vehicle inputs directly and have a steering function with a short prediction horizon. Usually far less than a second. Therefore, these methods require many samples (and thus path segments) to reach the goal. In contrast, the CL-RRT can generate path segments of several seconds. Therefore, this variant requires fewer samples to reach the goal and may suffer less from path meandering issues than traditional variants. Thus, we base our method on the CL-RRT. However, due to the piecewise linear nature of the controller inputs, the CL-RRT is not capable of accurately following the curved roads. This issue remains to be solved before it can be applied to high speed driving on structured roads.

B. Contributions

The main contribution of this paper is a path bending method that enhances Closed-Loop RRTs [17] for planning on curved roads, the CA-CL-RRT, which can be followed with a predefined velocity. Path planning is enhanced such that the planned path does not present undesired wandering around the lane center. We evaluate the performance of the proposed method with randomized highway driving, where we compare the introduced approach with RRT and CL-RRT. Further, we expand the Local Model Predictive Contouring Control (LMPCC) proposed in [9] to follow a recursively updated trajectory reference provided by our global planner, and present simulations results, comparing both methods in a dynamic collision avoidance scenario.

II. Problem formulation

We apply the following assumptions during the problem definition:

- Obstacle positions and vehicle states are known
- The goal state is provided by a mission planner
- Lane markings are provided by a lane detection system as coefficients of a 2nd order polynomial.
- A velocity profile is given to generate a trajectory from the computed path

A. Vehicle representation

The Autonomous Vehicle (AV) operates on a world plane $W \in \mathbb{R}^2$. The dynamics of the AV are represented by a set of nonlinear dynamic equations:

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)), \quad \mathbf{x}(0) = x_0 \tag{1}$$

where $\mathbf{x}(t) \in \mathbb{R}^{n_x}$ and $\mathbf{u}(t) \in \mathbb{R}^{n_u}$ are the vehicle states and inputs respectively and $\mathbf{x}(0)$ is the initial state of the vehicle. The body of the AV is modeled with a 2-Dimensional Oriented Bounding Box (2D-OBB) that is aligned with the vehicle heading. The region of the state-space that the body of the AV occupies is denoted as $\mathcal{A}(\mathbf{x}) \in X$.

B. Motion planning problem

The main objective is to find a feasible trajectory τ, consisting of a sequence of T states $\mathbf{x}_m = [x^r_m, y^r_m, \theta^r_m, v^r_m] \in X$ with $m \in \{1, \dots, T\}$, from an initial $\mathbf{x}(0)$ to a goal state region X_{goal}, that minimizes the lane center deviation, path curvature and path length, within the local car-coordinate frame (see Section IV-B.1). The upper-script r is used to denote the reference trajectory. More formally, we define the motion planning problem as follows:

$$\tau^* = \arg\min_{\tau} \quad \sum_{0}^{M} J(\tau_m) = (3) \tag{2a}$$

$$\text{s.t.} \quad (1) \tag{2b}$$

$$\mathcal{A}(\mathbf{x}) \in X \setminus (X_{\text{obs}} \cup X_{\text{dyn}}) \tag{2c}$$

$$\mathbf{u}(t) \in U \qquad \forall t \in [0, T] \tag{2d}$$

$$x(T) \in X_{\text{goal}} \tag{2e}$$

where U is the set of admissible inputs and $J(\cdot)$ is the trajectory cost function (defined in Section IV-B.1). X_{obs} is the region in the state-space that violates the collision constraints, and X_{dyn} the set of vehicle's states that violate stability criteria, dynamic limitations and actuator saturation (See Section IV-A.2). Future positions of dynamic obstacles are predicted with a constant velocity model.

III. Preliminaries

A. Trajectory Tracking Controller

Consider that a reference trajectory τ, consisting of a sequence of T reference points $\boldsymbol{\tau}^r_m = [x^r_m, y^r_m, \theta^r_m, v^r_m]$ with $m \in \{1, \dots, T\}$, is provided by a global planner (e.g., CA-CL-RRT). A trajectory tracking controller receives the reference trajectory τ and generates acceleration and steering

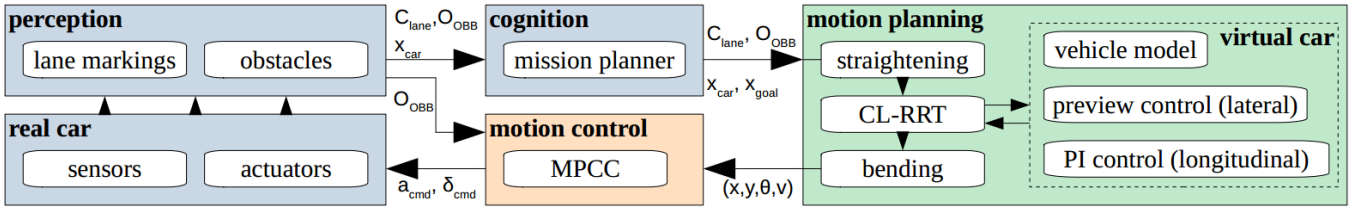


Fig. 2: Schematic overview of the proposed method: C_{lane}: lane marking coefficients, O_{OBB}: obstacle Oriented Bounding Box, x_{car}: vehicle states, x_{goal}: goal states, a_{cmd}: acceleration command, δ_{cmd}: steering angle command.

commands $\mathbf{u} = [a_{cmd}, \delta_{cmd}]$ to ensure that the vehicle follows the planned trajectory. In this work, we employ an MPCC formulation, more specifically the LMPCC approach proposed in [9], as a trajectory tracking controller for the following reasons. First, LMPCC enables one to follow a trajectory provided by a global planner and compute control commands for the vehicle. Second, LMPCC allows us to take into account the predicted paths of dynamic obstacles and to incorporate collision avoidance constraints. Hence, the LMPCC approach enables trajectory tracking and local collision avoidance if the global path is not computed in time to avoid dynamic obstacles. For more details, we refer the reader to [9], [18].

IV. Method

The overall system architecture is presented in Fig. 2. The vehicle is equipped with sensors to detect obstacles and lane markings. The mission planner uses this data and the vehicle state information to define a local goal sending it to the motion planner. Our motion planning method builds on [17] consisting of the following three steps:

1) Transform the planner inputs from the curved road to a virtual straightened road;
2) Build a tree towards the goal using the CL-RRT;
3) Bend the best motion plan back to the curved road.

Finally, we employ the motion controller introduced in the preliminaries that enables the vehicle to track the planned trajectory from the motion planner [9]. In this section, first, the CL-RRT method which we build on is presented. Then, in Section IV-B, the proposed adjustments applied to the CL-RRT enhancing its application to curved roads by applying the road transformations as described in Section IV-C. Last, the re-planning for dynamic collision avoidance is discussed in Section IV-E.

A. Closed-Loop Rapidly-exploring Random Tree

The CL-RRT incrementally builds a tree of feasible trajectories towards the goal. During each tree expansion step (see Algorithm 1), first a random sample is generated in the workspace (line 1). Tree nodes are then sorted using heuristics (line 2). Multiple nearest nodes are selected (line 3), which in turn, are expanded until a feasible trajectory is generated (line 5-9). This expansion is done through closed-loop prediction of the vehicle trajectory (see Section IV-A.1). If a new node is added, a goal biased expansion is attempted (lines 10-12). If this expansion is feasible (i.e., if satisfies Eq. (2c) and Eq. (2d)) (line 13), it is added to the tree (line 14). For more details on the algorithm, see [17].

Algorithm 1: ExpandTree($\mathcal{T}$) [17]

```
1  Get random sample s ∈ ℝ².
2  Sort nodes n using heuristics.
3  Select N_near nearest neighbors.
4  for all n ∈ N_near do
5      Form a reference command from n to s
6      Obtain a trajectory x(t) ∈ [t1, t2] by doing a
         simulation until the end of reference is reached.
7      if x(t) is feasible ∀t ∈ [t1, t2] then
8          T.add_node
9          break
10 if a node was added then
11     Form a reference command from s to goal.
12     Obtain a trajectory x(t) ∈ [t2, t3] by doing a
         simulation until the end of the goal reference is
         reached.
13     if x(t) is feasible ∀t ∈ [t2, t3] then
14         T.add_node
15 return
```

1) Closed-loop prediction: A virtual car is driven along a generated reference path and velocity profile (see Section IV-B.2) to obtain a closed-loop prediction of its trajectory. If the simulated trajectory satisfies constraints (2c) and (2d) we add it to the tree.

2) Vehicle model: We model the vehicle using the extended linear dynamic bicycle model as proposed in [17]. To ensure the vehicle stability and consistency of the linear bicycle model [19], we constrain the lateral ($a_{lateral} \leq 0.3g$) and longitudinal ($a_{longitudinal} \leq 1.5m/s^2$) acceleration during planning. Here, g is the Earth's gravitational acceleration constant.

B. Enhancing CL-RRT for highway driving

To enhance CL-RRT for application to high-speed driving on curved roads, we propose the following modifications to the original method.

1) Cost function: The cost function employed in the baseline planning method (CL-RRT [17]) minimizes path length. Consequently, this can result in corner cutting, which makes the cost function not suitable for a natural driving style. To realize a natural style that is also suitable for highway driving, we introduce the cost function in (3) that

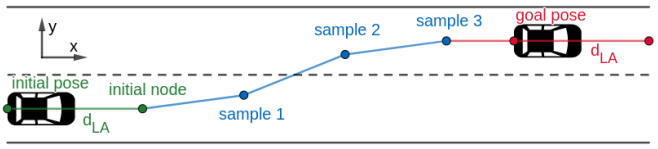


Fig. 3: Reference path generation. In green, initialization of the tree with a node at preview distance d_{LA}. Then in blue, several exploration expansions. Lastly in red, a goal biased expansion with look-ahead distance d_{LA}.

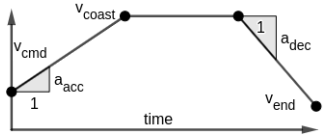


Fig. 4: Velocity profile

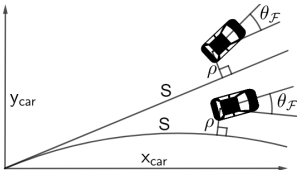


Fig. 5: Road straightening

penalizes path length, path curvature, and deviation from lane center.

$$J_{\text{CA-CL-RRT}} = \sum_{s=1}^{s=S} \sum_{i=1}^{i=I} \left(w_u u \Delta t + w_\kappa \kappa_{n,i} + w_D D_{n,i} \right) \quad (3)$$

where S is the number of path segments, I the number iterations used in the closed-loop predictions for the segment n, w_u, w_κ and w_D are the cost weights, u is the longitudinal velocity, Δt is the prediction step size, $\kappa_{n,i}$ is the curvature of the path, and $D_{n,i}$ is the distance to the goal lane.

2) Reference generation: We employ the reference generation strategy as proposed in [7], [17], which constructs a piece-wise linear path and designs a deterministic trapezoidal velocity profile. A goal-biased reference connects the sample with the goal position, and extends this with an extra segment that is aligned with the goal heading (see Fig. 3), and has the length of the look-ahead distance of the lateral controller (see Section IV-B.3). We employ the velocity profile proposed in [7], and introduce a variable end velocity (v_{end}), see Fig. 4:

$$\frac{v_{max}^2 - v_0^2}{2a_{acc}} + v_{max} t_{min} + \frac{v_{max}^2 - v_{end}^2}{2a_{dec}} < D \quad (4)$$

$$\frac{v_{coast}^2 - v_0^2}{2a_{acc}} + v_{coast} t_{min} + \frac{v_{coast}^2 - v_{end}^2}{2a_{dec}} = D \quad (5)$$

where, D is the path length. v_0, v_{coast} and v_{end} are the initial, coasting, and end velocity, respectively. v_{max} is the speed limit, and t_{min} is the minimum coasting time. Lastly, a_{acc} and a_{dec} are the acceleration and deceleration respectively. Here, the first, second and third terms represent the acceleration, coast and brake distance, respectively. We use these equations as follows. If (4) can be satisfied, we select $v_{coast} = v_{max}$. Otherwise, (5) is solved to obtain v_{coast}. Lastly, when $v_{end} > v_{coast}$, we select $v_{coast} = v_{end}$.

3) Controllers: We apply the single-point preview controller presented in [20], which considers a dynamic vehicle model and is suitable for high-speed driving. It uses a preview point ahead of the vehicle, of which the position depends on the look-ahead time and longitudinal velocity. We vary the look-ahead time to enhance maneuverability at low speeds and curvature smoothness at high speeds.

For longitudinal control we apply the PI controller as proposed in [17].

C. Road transformation

The piece-wise linear reference path, described in Section IV-B.2, cannot accurately follow the road curvature when motion is planned directly on the curved road. Hence, we propose to plan on a virtual straightened road and deform the generated plan motion plan to follow the shape of the road, inspired by the approaches proposed in [2], [3], [8]. The virtual straight road allows the planner to remain using piece-wise references while constructing references that accurately follow the road curvature. Additionally, this approach avoids corner-cutting issues caused by distance-based optimization heuristics. In the following paragraphs, we denote equations belonging to the straight and curved with superscripts $^{\mathcal{S}}$ and $^{\mathcal{C}}$, respectively.

1) Straightened road definition: We model the road as a second-order polynomial ${}^{\mathcal{C}}y_{cl}(x) = c_2x^2 + c_1x + c_0$, provided by a lane detection system. We use this to construct a virtual straightened road tangent to the curved road: ${}^{\mathcal{S}}y_{cl}(x) = c_1x + c_0$, as depicted in Fig. 5.

2) Arc-length parametrization: The transformation between the curved and straight road is realized through an intermediate Frénet frame (see Fig. 5). A pose in this frame is defined as $P_F = [S, \rho, \theta_F]$, where S is the position along the lane, ρ is the lateral displacement, and θ_F is the difference in the heading.

For the transformation between the frames, we need an arc-length parametrization $S(x)$ for both roads. Obtaining an analytical parametrization for the straight road is straightforward. In contrast, for the curved, road this involves solving a fifth-order root and is therefore fairly complex. Hence, we numerically approximate the parametrization with a second-order polynomial.

3) Transformation from curved to straight: Transforming a state ($\mathbf{x} \in \mathbb{R}^{n_x}$) from the curved to the straight road involves the transformation of a pose (x, y, θ) and the steering angle (δ). The remaining states $(v, a, \dot{a})$ do not change during the transformation. Firsts, we transform a pose $P_i = [x_i, y_i, \theta_i]$ to the straight road by evaluating equations (6-9) Then, we locate the closest point along the center-line of the curved road and determine the road heading at that point. We use this to calculate the relative Frénet frame pose (7). Finally, we calculate the heading of the straight road (8) and use it to transform the pose to the straight road (9).

$$P_c = \begin{bmatrix} x_c \\ y_c \\ \theta_c \end{bmatrix} = \begin{bmatrix} \text{minimize} \left\| \begin{bmatrix} x_i \\ y_i \end{bmatrix} - \begin{bmatrix} x \\ {}^{\mathcal{C}}y_{cl}(x) \end{bmatrix} \right\| \\ \arctan2\left(({}^{\mathcal{C}}y_{cl}(x)/dx)|_{x=x_c}, x_c \right) \end{bmatrix} \quad (6)$$

$$P_F = \begin{bmatrix} S \\ \rho \\ \theta_F \end{bmatrix} = \begin{bmatrix} {}^{\mathcal{C}}S(x_c) \\ \text{sign}(\theta_i - \theta_c) \left\| \begin{bmatrix} x_i \\ y_i \end{bmatrix} - \begin{bmatrix} x_c \\ y_c \end{bmatrix} \right\| \\ \theta_i - \theta_c \end{bmatrix} \quad (7)$$

$$\theta_{rs} = \arctan2(d^{\mathcal{S}} y_{cl}/dx, 1) \tag{8}$$

$$P_s = \begin{bmatrix} x_s \\ y_s \\ \theta_s \end{bmatrix} = \begin{bmatrix} \cos(\theta_{rs})S - \sin(\theta_{rs})\rho \\ \sin(\theta_{rs})S + \cos(\theta_{rs})\rho + c_0 \\ \theta_F + \theta_{rs} \end{bmatrix} \tag{9}$$

The only thing remaining is the transformation of δ, which is solved by subtracting the steering angle required for following the road center at its longitudinal coordinate S. We realized the transformation of a state in constant time by evaluating closed-form expressions. Transforming the planned motion back to the curved road can be done by applying the inverse of the discussed calculations.

D. Planning on the straight road

The proposed transformations facilitate the planning steps introduced at the beginning of Section IV. By using this method, we can plan on the virtual road as long as the lane detection system can provide the road parametrization. If this system fails to detect the road, we can skip the transformations and apply CL-RRT directly to the curved road. While bending the final motion plan we introduce additional curvature (κ), and thus lateral acceleration ($a_y = u^2\kappa$). This was included in the dynamic constraints (2c) by calculating an upper bound of the curvature. We calculate path curvature as $\kappa(x) = y''/(1+y'^2)^{\frac{3}{2}}$, and establish the upper bound by solving $d\kappa/dx = 0$.

E. Dealing with dynamic environments

To deal with dynamic obstacles, we re-plan continuously and incorporate trajectory prediction information about the dynamic agents. The predictions are obtained, assuming a constant velocity model. Algorithm 2 summarizes the main steps of the proposed CA-CL-RRT planning method. First, the path planner updates all inputs (line 1). Then, the inputs and the previously computed trajectory are transformed into the straight virtual road space (lines 2-3). The tree is initialized with the previous trajectory (line 4) and expanded while time is available (line 5-6). The new best trajectory is selected (line 7) and transformed back to the curved road (line 8-9). Lastly, the planned trajectory is sent to the controller for execution (line 10).

Algorithm 2: CA-CL-RRT: Curvature-Aware Rapidly-exploring Random Trees

1 Update $\mathbf{x}_{car}$, $\mathbf{x}_{goal}$, X_{obs}, $\mathcal{R}oad$
2 **if** $\exists \mathcal{R}oad$ **then**
3 Transform $\mathbf{x}_{car}$, $\mathbf{x}_{goal}$, X_{obs} and $\boldsymbol{\tau}_{old}$ from $\mathcal{C}$ to $\mathcal{S}$ according to Eq. (6-9)
4 $\mathcal{T}$.initialize($\boldsymbol{\tau}_{old}$)
5 **while** $t_0 < t_0 + \Delta t$ **do**
6 $ExpandTree(\mathcal{T})$ (see Algorithm 1)
7 Select $\boldsymbol{\tau}_{new}$ according to Eq. (3)
8 **if** $\exists \mathcal{R}oad$ **then**
9 Transform $\boldsymbol{\tau}_{new}$ from $\mathcal{S}$ to $\mathcal{C}$ with inverse of Eq. (6-9)
10 Apply $\boldsymbol{\tau}_{new}$. Set $\boldsymbol{\tau}_{old} = \boldsymbol{\tau}_{new}$.

TABLE I: Vehicle parameters

Parameter	Value	Unit
δ_{max}	0.52	rad
$\dot{\delta}_{max}$	0.3294	rad/s
T_d	0.3	$-$
T_a	0.3	$-$
a_{min}	-6	ms^2
a_{max}	2	ms^2
L	2.7	m
K_{us}	0.014	$-$
ρ	5.95	m
w_{body}	2	m
L_{body}	4.7	m
l_r	1	m

TABLE II: Planner parameters

Parameter	Value	Unit
a_{acc}	1	ms^2
a_{dec}	-1	ms^2
t_{min}	1	s
$a_{y,max}$	0.3	g
$t_{la,highway}$	1.4	s
K_p	4	$-$
K_i	0.05	$-$
P_{exp}	0.7	$-$
P_{opt}	0.3	$-$

V. Simulation results

The proposed method is evaluated in urban and highway environments with static and dynamics obstacles (e.g., pedestrians or other human drivers):

- Section V-B: Highway with static obstacles
- Section V-C: Dynamic obstacle avoidance

A. Simulation environment

The simulations were performed on a mobile workstation with an Intel i5 CPU@2.6GHz. The highway scenario was implemented in MATLAB R2019b and simulates the RRT planners only. For the dynamic collision avoidance scenario, we implemented the full planning architecture, described in Fig. 2, in C++ optimized for real-time performance and ROS as middleware. Additionally, we use the open-source implementation of [9] for the pedestrian's simulation scenario and LMPCC implementation for the motion controller. The motion planner (CA-CL-RRT) and controller (LMPCC) have an update rate of 5Hz and 50Hz, respectively. The vehicle parameters used during simulation are defined in Table I. Here, w_{body} and L_{body} are the width and length of the vehicle body. l_r is the distance from the rear of the vehicle body to the rear axle coordinate frame. The motion planner configuration parameters are defined in Table II.

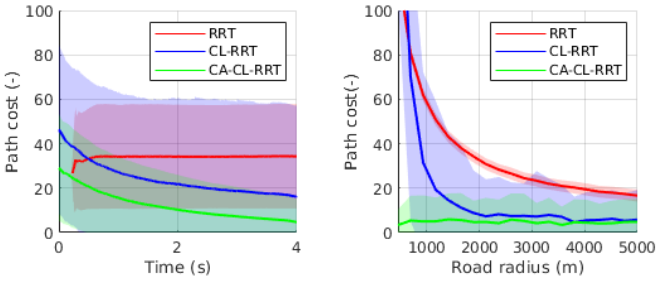
B. Highway driving with static obstacle

The bending method employs a second-order polynomial and can plan on any road that can be captured with this. It can therefore plan on relatively simple geometries, like highways, which are designed with guidelines that ensure curvature limits. Here, we consider a highway scenario with constant curvature and two lanes. We test two maneuvers: lane following and lane changing. Fig. 1 shows the considered scenario. Notice here, that the tree was built on the virtual straightened road, as depicted in the bottom subplot of Fig. 1. Only the selected trajectory is deformed to the curved road.

1) Compared algorithms: The proposed method (CA-CL-RRT) is compared with two baseline approaches: the standard RRT and CL-RRT to plan directly on the curved road. All algorithms employ the same optimization heuristic and cost function (3), with weights $w_u = 0.01$, $w_\kappa = 0.01$, $w_D = 100$.

TABLE III: Statistic results of motion planning for a constant curvature highway for $R_n \in [450, 5000], n = 1...20$. The radii are linearly distributed. Each algorithm plans 100 times for each road configuration. Considered are 0 and 1 obstacles (randomized location for each query), and scenarios Lane Following (LF) and Lane Change (LC).

config.	% failures			Final cost: mean (Std)			Samples explored: mean (Std)			First goal comp. time: mean (Std)		
	RRT	CL RRT	CA-CL RRT	RRT	CL RRT	CA-CL RRT	RRT	CL RRT	CA-CL RRT	RRT	CL RRT	CA-CL RRT
0 obs, LF	13.8	6.0	**0.0**	32 (25)	9 (31)	**1** (2)	351 (27)	35 (6)	35 (7)	0.8 (0.7)	0.5 (0.5)	0.6 (0.3)
1 obs, LF	23.5	18.8	**1.2**	33 (23)	23 (46)	**10** (15)	330 (35)	20 (5)	22 (6)	1.2 (0.9)	1.2 (1.0)	1.2 (0.8)
0 obs, LC	19.4	0.1	**0.0**	36 (23)	10 (37)	**1** (3)	376 (20)	37 (6)	39 (7)	0.8 (0.8)	0.3 (0.3)	0.4 (0.2)
1 obs, LC	28.4	1.5	**0.3**	35 (22)	24 (48)	**7** (11)	363 (28)	20 (4)	23 (6)	1.1 (0.9)	0.9 (0.7)	1.0 (0.7)



a) Cost vs computation time *b)* Cost vs radius

Fig. 6: Highway comparison results

The standard RRT uses a kinematic bicycle model without actuator constraints. A predefined discrete set of steering angles U is employed. During node expansion, all steering inputs $u_i \in U$ are simulated, which leads to many short path segments. The segment that ends closest to the randomly sampled location is added to the tree. The corresponding u_i is removed from this node's set U.

During simulations we employ $U = u_i \in [-\delta_{max}, \delta_{max}]$, for $i = \{1, 2, ..., 11\}$, and $\delta_{max} = 0.0312$ radians. The simulation horizon is set to 0.25 seconds.

2) Road configurations: The goal position is provided by a mission planner (See Fig. 2) and is located 150 meters ahead of the ego vehicle and can be on either of the lanes. The vehicle has a preferred speed of 120km/h. In total, 20 roads are considered, with the radii linearly distributed between 450 and 5000 m. Each algorithm plans motion 200 times for each road configuration. Half of these queries have an obstacle that is randomly placed on either of the lanes. We performed 4000 motion queries per algorithm and constrained the computation time per query to 4 seconds.

3) Results: Fig. 6a shows the evolution of the path cost as computation time elapses, and Fig. 6b shows the cost of the selected path versus the road radius. The presented results in these figures are calculated using only motion queries that successfully reach the goal. The bold lines represent the means and the shaded areas the standard deviation.

Additional results are presented in Table III. In this table, the 1.2% failure rate of the proposed method could be caused by obstacles being placed too close to the ego vehicle, making avoidance impossible given the proposed lateral acceleration constraint.

C. Dynamic obstacle avoidance

We compare the performance of our planner with the LMPCC planner [9] as a baseline. The LMPCC was set up to a prediction horizon of 3 s, with 15 stages and considering the middle lane as a reference path. In contrast, as the reference path generated by our method already accounts for the predictions of the pedestrians, we reduced the prediction horizon to 1 s for the trajectory tracking controller, as depicted in Fig. 2. Fig. 7 depicts the setup of the experiment where the autonomous vehicle has to navigate along a straight road while avoiding two pedestrians that will cross the road at random times and from both sides. The statistical results presented in Table IV show that the proposed method matches the collision avoidance performance of the baseline approach despite the reduced prediction horizon. The simulations that resulted in a collision are due to the model mismatch used in the tracking controller and the real vehicle and correspond to situations where the braking command was not strong enough to stop the car in time. Yet, the approach allows us to significantly reduce the solver computations times, allowing, as future work, to incorporate a higher fidelity vehicle model to improve the overall collision avoidance performance.

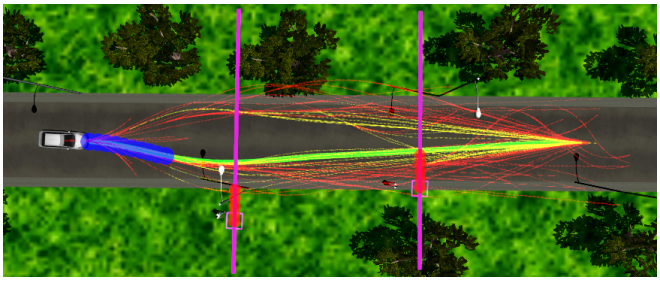


Fig. 7: Pedestrian collision avoidance scenario. Red depicts the feasible trajectories, in yellow the feasible trajectories that reach the goal, in green the selected trajectory, in purple the pedestrian paths, the blue circles the trajectory of the AV and the red circles the predicted pedestrian trajectories.

VI. Discussion

The highway simulation results presented in Fig. 6a reveal that the proposed method plans paths with a significantly lower cost compared with the two baselines. Moreover, it also shows that while CL-RRT and CA-CL-RRT improve the path quality over time, the RRT quickly gets stuck in its initial solution. Results have shown that the proposed method can accurately track the lane centerline without any meandering.

TABLE IV: Statistic results of percentage number of collisions, time to reach final destination, and solver mean computation times for 100 random test cases of the dynamic collision avoidance experiment with two pedestrians. The pedestrians follow the social forces model [21].

Method	% Collisions	Time to goal [s]	Mean Computation Time [ms]
CA-CL-RRT	5	10.47	0.9
LMPCC	6	9.5	10.1

Fig. 6b demonstrates the advantage of our method. On curved roads, the proposed method plans paths of significantly lower cost compared to RRT and CL-RRT. Yet, the CL-RRT cost converges to CA-CL-RRT as the radius increases, which can be easily explained by the fact that the virtual straightened road varies less from the real road. Thus, CL-RRT essentially becomes the same planner as the CA-CL-RRT. The high cost of CL-RRT on curved roads can be explained by the piecewise linear nature of the reference path, which makes it extremely hard to follow the road centerline accurately without meandering.

Considering that the Dutch directive for highway design states that highway radii must be at least 750 meters [22], we conclude that our approach significantly improves motion planning on roads with realistic curvature in comparison with the two baselines.

Table III shows that CA-CL-RRT has fewer planning failures and produces paths with lower cost on the considered curved roads. Furthermore, the CL-RRT and CA-CL-RRT explore significantly fewer samples than the RRT since a single sample leads to a significantly longer path section than the RRT.

VII. Conclusion

This article presents a motion planning framework, the CA-CL-RRT, enhancing path quality in structured environments with curved roads. We present qualitative and quantitative results against two baselines: RRT and CL-RRT in a highway driving scenario. Results have shown that the proposed CA-CL-RRT can significantly enhance the path quality, especially in curved roads with a radius of less than 1000 m. We combined the CA-CL-RRT with LMPCC and compared it with LMPCC as a baseline during a dynamic pedestrian avoidance simulation. The combination significantly reduced the required computational effort of the LMPCC.

Same as CL-RRT baseline method, the presented motion planner is also using a deterministic way for exploration of the velocity dimension. Furthermore, the algorithm tends to produce sub-optimal paths in complex environments, which arises from the sub-optimal nature of the RRT. To address the identified shortcomings, future research could focus on enhancing the velocity space exploration and introducing asymptotic optimality.

References

[1] J. Ziegler and C. Stiller, "Spatiotemporal state lattices for fast trajectory planning in dynamic on-road driving scenarios," *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009*, pp. 1879–1884, 2009.

[2] M. Werling, J. Ziegler, S. Kammel, and S. Thrun, "Optimal trajectory generation for dynamic street scenarios in a frenét frame," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 987–993, 2010.

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

[4] K. Zheng and S. Liu, "RRT based Path Planning for Autonomous Parking of Vehicle," *2018 IEEE 7th Data Driven Control and Learning Systems Conference (DDCLS)*, pp. 627–632, 2018.

[5] F. Esposto, J. Goos, A. Teerhuis, and M. Alirezaei, "Hybrid path planning for non-holonomic autonomous vehicles: An experimental evaluation," in *5th IEEE International Conference on Models and Technologies for Intelligent Transportation Systems, MT-ITS 2017 - Proceedings*, 2017, pp. 25–30.

[6] D. Ferguson, T. Howard, and M. Likhachev, "Motion Planning in Urban Environments," *Journal of Field Robotics*, vol. 25, 2008.

[7] Y. Kuwata, G. Fiore, J. Teo, E. Frazzoli, and J. How, "Motion planning for urban driving using RRT," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*. IEEE, 9 2008, pp. 1681–1686. [Online]. Available: http://ieeexplore.ieee.org/document/4651075/

[8] M. Rufli and R. Siegwart, "On the design of deformable input- / state-lattice graphs," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 3071–3077, 2010.

[9] B. Brito, B. Floor, L. Ferranti, and J. Alonso-Mora, "Model Predictive Contouring Control for Collision Avoidance in Unstructured Dynamic Environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4459–4466, 7 2019.

[10] S. M. LaValle, "Rapidly-Exploring Random Trees: A New Tool for Path Planning," Tech. Rep., 1998.

[11] S. Karaman and E. Frazzoli, "Incremental Sampling-based Algorithms for Optimal Motion Planning," *International Journal of Robotics Research*, 5 2010. [Online]. Available: http://arxiv.org/abs/1005.0416

[12] S. Lavalle, *Planning Algorithms*. Cambridge: Cambridge University Press, 2006.

[13] C. Urmson and R. Simmons, "Approaches for Heuristically Biasing RRT Growth," in *IEEE International Conference on Intelligent Robots and Systems*, vol. 2, 2003, pp. 1178–1183.

[14] E. Frazzoli, M. A. Dahleh, and E. Feron, "Real-Time Motion Planning for Agile Autonomous Vehicles," *Journal of Guidance, Control, and Dynamics*, vol. 25, no. 1, pp. 116–129, 2002. [Online]. Available: http://arc.aiaa.org/doi/10.2514/2.4856

[15] K. Yang, "An efficient Spline-based RRT path planner for non-holonomic robots in cluttered environments," *2013 International Conference on Unmanned Aircraft Systems, ICUAS 2013 - Conference Proceedings*, pp. 288–297, 2013.

[16] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli, and S. Teller, "Anytime Motion Planning using the RRT*," *IEEE Conference on Robotics and Automation*, 2011.

[17] Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli, and J. P. How, "Real-time motion planning with applications to autonomous urban driving," *IEEE Transactions on Control Systems Technology*, vol. 17, no. 5, pp. 1105–1118, 2009.

[18] L. Ferranti, B. Brito, E. Pool, Y. Zheng, R. M. Ensing, R. Happee, B. Shyrokau, J. F. Kooij, J. Alonso-Mora, and D. M. Gavrila, "SafeVRU: A research platform for the interaction of self-driving vehicles with vulnerable road users," in *IEEE Intelligent Vehicles Symposium, Proceedings*, vol. 2019-June, 2019, pp. 1660–1666.

[19] P. Polack, F. Altche, B. DAndrea-Novel, and A. De La Fortelle, "The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?" in *IEEE Intelligent Vehicles Symposium, Proceedings*, 2017, pp. 812–818.

[20] A. Schmeitz, J. Zegers, J. Ploeg, and M. Alirezaei, "Towards a generic lateral control concept for cooperative automated driving theoretical and experimental evaluation," in *5th IEEE International Conference on Models and Technologies for Intelligent Transportation Systems, MT-ITS 2017 - Proceedings*. Institute of Electrical and Electronics Engineers Inc., 8 2017, pp. 134–139.

[21] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," *Physical review E*, vol. 51, no. 5, p. 4282, 1995.

[22] Rijkswaterstaat (Ministerie van Infrastructuur en Milieu), "Richtlijn Ontwerp Autosnelwegen 2019," vol. 21, no. november, 2019.