F1TENTH: Autonomous overtaking

Olivia Aschemann
M.Sc. Mechatronics, Robotics, and Biomech. Eng.

Weiqi Lyu
M.Sc. Mechanical Engineering

Yanwen Liu
M.Sc. Mechatronics and Robotics

***Abstract*—Overtaking scenarios are considered one of the most complex tasks for autonomous driving, because it requires real-time decision-making and trajectory planning. This paper integrates Rapidly Exploring Random Trees (RRT) with Stanley avoidance-based PID control to facilitate overtaking maneuvers demonstrated on the F1TENTH racing car. Experimental results indicate that employing this combined algorithm enables the vehicle to successfully navigate through obstacles and execute overtaking maneuvers. Compared to the RRT algorithm, the vehicle demonstrates improved lap times on the track.**

***Index Terms*—autonomous overtaking, RRT, Stanley control, PID, autonomous vehicle, F1TENTH**

I. Motivation

In traffic safety, the overtaking scenario is considered one of the most complex driving maneuvers out of several driving maneuvers, such as lane changing, lane following, tailgating, and many more. It has multiple challenges, including advanced perception and prediction, decision-making, coordination to safely navigate dynamic traffic environments and risk assessment. Rapidly Exploring Random Trees (RRT) can overcome some of these challenges by rapidly exploring the solution space, allowing real-time decision-making, and reacting to the dynamic environment.

II. State of the art

A. Classic methods

1) Sampling-based methods: Kala and Warwick [1] used RRT for planning multiple vehicles in traffic scenarios. The team at the University of Waterloo [7] successfully deployed this algorithm on the F1TENTH (F1/10) racing car and adjusted its parameters accordingly.
2) Model Predictive Control (MPC): Németh [2] designed an MPC-based method for overtaking maneuvers of autonomous vehicles. The road and velocity profile of the autonomous vehicle could be determined through a graph-based optimization algorithm, which is guaranteed by the proposed MPC control. The novel algorithm can consider multi-vehicle scenarios during overtaking maneuvers. Ruchi [7] designed a model race car in ROS2 using MPC with a kinematic bicycle model and a global optimal trajectory.

B. AI-based methods

1) Reinforcement learning: Jia Liu [3] designed a task-level sequencing curriculum reinforcement learning method to tackle reward sparseness. It breaks down a complex task into two successive subtasks: the first aims to drive as fast as possible in a single-vehicle model, and the second aims to overtake other vehicles without any collisions according to the knowledge gained from the first task in a multi-vehicle model. In the simulator, the overtaking time and distance were reduced by 8.7% and 4.9%, respectively.
2) End-to-end learning: this method aims to directly map sensory inputs to control outputs by passing intermediate processing stages. Tianhao Wu [4] proposed an end-to-end driving model based on a Long Short-Term Memory neural network, which aims to encode the spatiotemporal information from different scales for steering angle prediction. Experimental results showed that the proposed driving model can predict the steering angle with high accuracy in complex scenarios.

III. Methodical approach

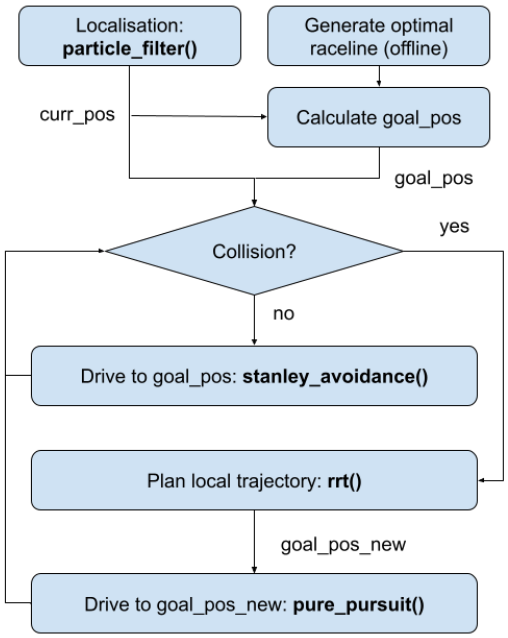

Fig. 1. Implemented RRT-Stanley-algorithm

The global path (optimal raceline) generated by the Race-Optimization repository [8] serves as our baseline.

For localization, we employed a particle filter [7]. To track the waypoints of the raceline, we utilize the Stanley-Avoidance package [7], integrating a PID controller, Stanley controller, and Pure Pursuit. Additionally, our local planner relies on RRT. The high-level idea of our implementation is shown in Fig. 1.

To simplify the overtaking process, we make the following assumptions about the opponent vehicle (OV): it maintains a constant low velocity and follows a smooth path without lane switching or blocking maneuvers. Given that the ego vehicle (EV) will outpace the OV, there is no need to accelerate excessively during overtaking. Additionally, predicting the OV's behavior becomes unnecessary in this scenario. Instead, by using a larger safety margin, we can regard the slow-driving OV (dynamic obstacle) as a static obstacle.

A. Raceline Generation

For the raceline generation, we use the Raceline-Optimisation repository [8], which is specifically tailored to the F1/10 car. This process is divided into the following steps:

1) Global map generation: use the slam toolbox to generate an occupancy grid map (Fig. 2) and refine (clean) the generated map using GIMP.

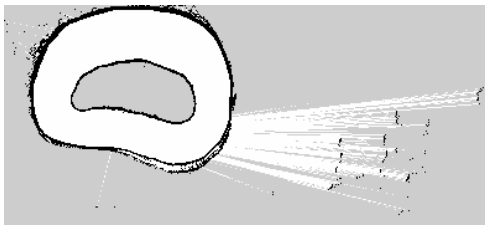

Fig. 2. Generated map with SLAM

2) Centerline extraction: use Euclidean distance transform to generate an orbit centerline.
3) Trajectory optimization: to facilitate overtaking maneuvers from both sides, we utilize the principle of minimum curvature [5] and calculate a conservative estimation of the track width, as shown in Fig. 3. This approach ensures an adequate space margin for overtaking scenarios while optimizing maneuverability on the track.

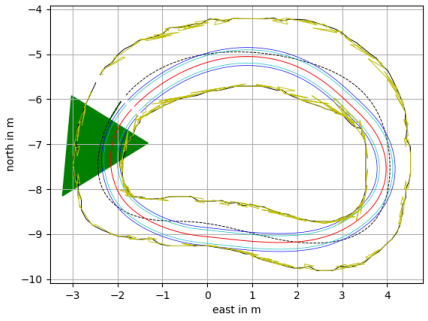

Fig. 3. Raceline with minimum curvature. The black dashed line is the centerline of the track. The red line is the corresponding optimal raceline.

B. Local planner

We adapted RRT as our local path planner to ensure real-time operation and effectively avoid obstacles. The workflow of the entire local path planning is the following:

1) Particle Filter: determine current car position.
2) Generate occupancy grid: the current occupancy grid will be generated for each lidar scan to determine whether obstacles occupy the current local map. The height of the occupation grid is determined by the lookahead distance d (LAD).
3) Goal position: determine the goal position by setting a point at a distance d from the current position along the raceline.
4) Check collision: RRT is activated if an obstacle occupies the space between the current position and the target position.

C. Controller

To enable the car to complete the race at a faster speed while overtaking and avoiding obstacles, we adopted two different control strategies:

1) Stanley-avoidance control: This strategy combines Stanley control (i.e., front wheel feedback control) with PID control. It enables the car to accurately track the global optimal raceline, ensuring fast race completion.
2) Pure Pursuit control: When RRT is activated, this control method tracks the newly generated local path points to achieve local obstacle avoidance and overtaking.

IV. Project Plan

To achieve our final goal, we defined the following intermediate steps:

1) mapping and generation of optimal raceline for the new track (offline) (week 1)
2) implementation of combined Stanley-RRT-algorithm (week 2)
3) fine-tune the parameters of the PID controller to follow the optimal raceline (week 3-4)
4) fine-tune the RRT parameters for static obstacles placed directly in the optimal raceline (week 3-4)
5) replacing the static obstacles with a slow-driving OV (week 4)
6) iteratively increasing the speed of the OV and EV (week 4)

V. Results

The best-performing parameters identified during tuning are demonstrated in Table I. We conducted two additional experiments to evaluate the performance of our combined RRT-Stanley algorithm compared to using RRT and Stanley independently. The experiments involved measuring the lap times under two conditions: a free track and a track with three boxes obstructing the car's raceline. The lap times for each condition are presented in Tables II and III, respectively.

The results show that our algorithm outperforms RRT in terms of speed, but is slower than the Stanley-avoidance

algorithm. One notable advantage of our algorithm is its ability to navigate around obstacles, a capability lacking in the Stanley algorithm.

Comparing our algorithm to RRT, we observe similar performance with significantly faster lap times. Nonetheless, RRT exhibits slightly fewer collisions than our algorithm. We explore potential explanations for these differences in performance in section VI.

TABLE I
PARAMETER FOR RRT-STANLEY WITH BEST PERFORMANCE

Parameter	Value
K_P	3.5
K_I	0.01
K_D	0.05
$K_{P\_obstacle}$	0.8
K_E	1.0
K_H	0.0
$cross\_track\_error\_offset$	0.1
$d_{min}[m]$	1.5
$d_{max}[m]$	1.75
$V_{d_{min}}[\frac{m}{s}]$	0.5
$V_{d_{max}}[\frac{m}{s}]$	1.0
$V_{min}[\frac{m}{s}]$	0.5
$V_{max}[\frac{m}{s}]$	1.0
$V_{percentage}[\%]$	0.25
$V_{percentage_{rrt}}[\%]$	0.25

TABLE II
LAP TIME (5 LAPS) WITHOUT OBSTACLES

Algorithm	Time [min:sec]
RRT [7]	1:57
RRT-Stanley	1:14
Stanley [7]	0:54

TABLE III
LAP TIME (5 LAPS) WITH THREE STATIC OBSTACLES

Algorithm	Time [min:sec]	Total collisions
RRT [7]	2:14	1
RRT-Stanley	1:26	2

VI. DISCUSSION

Adjusting the Look Ahead Distance (LAD) is a trade-off between optimizing the performance of the Stanley and RRT algorithms, respectively. Increasing the LAD causes the car to switch to RRT more frequently in corners where track boundaries intersect the straight line from the current to the target position, falsely triggering obstacle detection. Conversely, reducing the LAD delays object detection and limits the car's reaction time after switching to RRT for local trajectory planning, potentially leading to collisions.

Moreover, the implemented algorithm encounters difficulties in recognizing objects positioned along the sides, particularly the outer side, of the optimal trajectory. These objects often go unnoticed since they do not intersect the direct line between the current and target positions. Similarly, when two cars travel side by side, the car fails to detect any objects and attempts to return to the optimal raceline. Consequently, it cuts off the path of the other car, resulting in collisions in the majority of cases. To overcome this problem, it would be necessary to predict the OV behavior, or its velocity to adjust the path accordingly.

REFERENCES

[1] R. Kala and K. Warwick, "Planning of multiple autonomous vehicles using rrt," 2011 IEEE 10th International Conference on Cybernetic Intelligent Systems (CIS), 2011.

[2] B. Németh, T. Hegedűs and P. Gáspár, "Model predictive control design for overtaking maneuvers for multi-vehicle scenarios," 2019 18th European Control Conference (ECC), 2019.

[3] J. Liu, H. Li, Z. Yang, S. Dang and Z. Huang, "Deep Dense Network-Based Curriculum Reinforcement Learning for High-Speed Overtaking," in IEEE Intelligent Transportation Systems Magazine, vol. 15, no. 1, pp. 453-466, Jan.-Feb. 2023

[4] T. Wu, A. Luo, R. Huang, H. Cheng and Y. Zhao, "End-to-End Driving Model for Steering Control of Autonomous Vehicles with Future Spatiotemporal Features," 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 2019

[5] A. Heilmeier, A. Wischnewski, L. Hermansdorfer, J. Betz, M. Lienkamp, and B. Lohmann, "Minimum curvature trajectory planning and control for an autonomous race car," Vehicle System Dynamics, 2019

[6] Kausik Sivakumar, *Adaptive Planning with Model Predictive Control*. GitHub Repository, 2024, https://github.com/kausiksivakumar/Adaptive-Planning-with-Model-Predictive-Control.git, accessed on February 14, 2024.

[7] CL2-UWaterloo, *f1tenth_ws*, GitHub Repository, 2024, https://github.com/CL2-UWaterloo/f1tenth_ws.git, accessed on February 14, 2024.

[8] CL2-UWaterloo, *Raceline-Optimization*, GitHub Repository, 2024, https://github.com/CL2-UWaterloo/Raceline-Optimization.git, accessed on February 14, 2024.