ISSN 2959-6157

A Research on Single-Vehicle Trajectory Planning Algorithm in Multi-Obstacle Environments

Jiaxi Zhong

University of Electronic Science and Technology of China,Chengdu,610000,China;
18113956769@163.com

Abstract:

This work aims to investigate the optimal trajectory planning for single-vehicle scenarios in obstacle-filled environments. The first section of the paper provides background information and an overview of autonomous driving trajectory planning, emphasizing its crucial role in ensuring safety and efficiency. A review of previous literature shows a range of approaches and constraints for trajectory optimization, especially in complicated and dynamic situations. Then, the single-vehicle trajectory planning optimum control problem (OCP) is stated and, in order to solve it practically, it is transformed into a nonlinear programming (NLP) issue. A classification of numerical solution methods is presented, including direct and indirect approaches. Extensive trials conducted with different initial and terminal circumstances and with single and multiple obstacles have demonstrated that the model can reliably produce the desired outcomes. As a result, the study demonstrates the robustness and efficiency of the proposed algorithm.

Keywords: Trajectory Planning; Single-Vehicle; Optimal Control Problem; Full Discretization; Nonlinear Programming (NLP).

1. Introduction

Since China's Ministry of Industry and Information Technology declared its support for the commercial implementation of L3 autonomous driving features in 2023, autonomous driving technology has emerged as a major trend in transportation and has the potential to significantly change how we commute and transport goods. Autonomous driving has three primary sections, including perception, planning, and control. The planning phase is one of the most important of these stages because it creates the link between perception and control by ensuring the vehicle is on a safe, practical and effective route. Trajectory planning can be further categorized into single-vehicle and multi-vehicle planning. This paper focuses on single-vehicle trajectory planning, aiming to contribute to the advancement of this fundamental aspect of autonomous driving technology.

1.1 Development of Trajectory Planning for Autonomous Vehicles

Researchers have conducted various studies in the trajectory planning field. For example, Li et al. adopted vehicle-to-infrastructure (V2I) communication and traffic signal optimization technology to improve safety and efficiency at junctions and merging roadways[1]. Yesilyurt, Tunc, and Soylemez solved the problem of traffic congestion and energy consumption at junctions. They shared arrival times and optimized reservations using a multi-agent system with Vehicle Agents (VAs) and an Intersection Agent (IA). This method reduces waiting time and energy consumption while preventing collisions[2]. Bashir and Fleming employed a platoon-based strategy to solve the problem of coordinating autonomous vehicles through intersections. This approach allows only one platoon to be in the conflict zone at a time to ensure safety. Compared with the traditional stop sign control method, this approach reduces average delay per vehicle and communication overhead[3].

Hausknecht, Au, and Stone also contributed to optimizing traffic flow through several junctions. Their method employs Autonomous Intersection Management (AIM) with a multi-agent system, showing a significantly higher efficiency than conventional traffic controls[4]. Li and colleagues focused on optimizing trajectories for Connected and Automated Vehicles (CAVs) at unsignalized junctions. They used a decentralized initial guess framework and a centralized optimum control problem (OCP) method with numerical solutions. As a result, they can optimize CAV cooperation and manage junction traffic more effectively[5].

To prevent accidents in lane-free and multi-agent autonomous driving scenarios, Dimitrios Troullinos and colleagues used coordination graphs (CGs) and the max-plus method with artificial potential fields. Their approaches enable cars to reach appropriate speeds and further improve traffic flow[6]. Li et al. addressed the issue of fault-tolerant motion planning for CAVs at signal-free and lane-free junctions in their study. They proposed solving three parallel optimization problems to ensure vehicle safety under fault situations[7].

Through the study of Müller, Carlson, and Kraus, the problem of vehicle delay at intersections was solved. Their method schedules vehicle arrivals by using a Mixed Integer Linear Programming (MILP) technology, allowing them to efficiently manage traffic by optimizing arrival times and ensuring collision-free passage[8]. H. Wei, L. Mashayekhy, and J. Papineau adopted a game-in-game framework that integrated Vehicle-to-Vehicle (V2V) and V2I technologies, which was used to tackle the coordination of linked cars at signal-free intersections. Their approach not only improved junction throughput, but reduced accidents by 99%[9].

The last research was conducted by Bai Li, Youmin Zhang, and their colleagues, who discussed the vehicle coordination issue at unsignalized crossings. They used a batch-processing framework that combines reservation-based and planning-based techniques, thus greatly improving junction performance and safety.[10].

The above research demonstrates various innovative methods in autonomous driving trajectory planning. Based on these studies, we have developed a new approach to further improve the efficiency and safety of trajectory planning for autonomous vehicles.

1.2 Introduction Overview

Our study includes single-vehicle trajectory planning based on Optimal Control Problem (OCP), while incorporating real-time obstacle avoidance. The proposed algorithm constantly optimizes the motion of the vehicle and dynamically modifies the route to avoid obstacles. As a result, our works provide a safety foundation for the operation of autonomous driving in complex environments.

2. Optimal Control Problem for Single-Vehicle Trajectory Planning

The single-vehicle trajectory planning problem can be modeled as an optimal control problem (OCP). The following subsections(2.1-2.4) detailed the vehicle's kinematic constraints, boundary conditions, collision avoidance constraints and objective function respectively. The overall formulation of the OCP will be described in 2.5.

2.1 Vehicle Kinematic Constraints

The five equations of the vehicle's kinematic constraints are as follows:

$$\frac{dx(t)}{dt} = v(t) \cdot cos\theta(t)$$

$$\frac{dy(t)}{dt} = v(t) \cdot sin\theta(t)$$

$$\frac{dv(t)}{dt} = a(t)$$

$$\frac{d\phi(t)}{dt} = \omega(t)$$

$$\frac{d\theta(t)}{dt} = \frac{v(t) \cdot tan\phi(t)}{L}$$

Table 1. Parameters of vehicle kinematic constraints.

Sign of parameter(s)	The parameter(s) name
x, y	The vehicle's position in the 2D plane
θ	Orientation angle
v	Velocity
ϕ	Steering angle of the front wheels,
a	Acceleration
L	Wheelbase of the vehicle

These kinematic constraints are the basis of vehicle motion, and they are the basic conditions for trajectory planning problems.

Moreover, there are some physical limitations during the trajectory optimization process, which ensure that the solution is feasible and safe. The following dynamic constraints are imposed:

Velocity bounds: $v_{min} \leqslant v \leqslant v_{max}$, where $v_{min} = 0$ m/s, $v_{max} = 6$ m/s.

Acceleration bounds: $a_{min} \leqslant a \leqslant a_{max}$, where $a_{min} = -2$

m/s^2, $a_{max} = 2 \ \ m/s^2$.

Steering angle bounds: $\phi_{min} \leqslant \phi \leqslant \phi_{max}$, where $\phi_{min} = -\frac{\pi}{4}$, $\phi_{max} = \frac{\pi}{4}$.

Angular velocity bounds: $w_{min} \leqslant w \leqslant w_{max}$, where $w_{min} = -0.5$ rad/s, $w_{max} = 0.5$ rad/s.

2.2 Boundary Conditions

Boundary conditions are set at both ends of the trajectory, as Table 2 shows.

Table 2. Parameters of boundary conditions.

	Coordinates	Velocity	Orientation angle	Steering angle
Initial condition	$(x_0,\ y_0) = (0, 0)$	$v_0 = 3$	$\theta_0 = 0$	$\phi_0 = 0$
Final condition	$(x_f,\ y_f) = (15, 5)$	$v_f = 3$	$\theta_f = 0$	$\phi_f = 0$

It should be noted that once the car stops, the steering angle must be 0 to prevent damage to the vehicle.
These boundary conditions determine the beginning and ending points of the vehicle's trajectory, guiding the vehicle to find feasible paths that satisfy these constraints throughout the optimization process.

2.3 Collision Avoidance Constraints

Avoiding collisions with outside obstacles is a crucial part of trajectory planning. In this study, obstacles are represented by their circumscribed circles, regardless of their original shapes (e.g., circles, squares, or triangles). The vehicle is guaranteed to maintain a safe distance from the obstacle's edge by the collision avoidance restriction.

For an obstacle with center coordinates (x_{obs}, y_{obs}) and radius r_{obs}, assuming that the vehicle is approximated by a circle with radius $r_{vehicle}$, the collision avoidance constraint is expressed as:

$$\sqrt{(x - x_{obs})^2 + (y - y_{obs})^2} \geqslant r_{obs} + r_{vehicle}$$

, ensuring that the vehicle never crosses over into the obstacles. It is possible to include multiple obstacles in the model, each with a unique constraint.

2.4 Objective Function

For trajectory planning problems, the purpose of an objective function is to minimize the total energy consumption of vehicle control and ensure the smoothness of the trajectory. In our research, the objective function is defined in the following integral form:

$$F = \int_0^{tf} \left(a(t)^2 + \omega(t)^2 \right) dt$$

Table 3. Parameters about the objective function.

Sign of parameter	The parameter name
F	Objective function
tf	Total planning time
a(t)	Vehicle's acceleration at time t
w(t)	Vehicle's angular velocity at time t.

The above equation shows that the objective function is equal to the sum of squared acceleration and angular velocity. By setting the objective function, the optimization process can produce an economical and smooth trajectory. Therefore, the vehicle will follow an optimal path while meeting dynamic restrictions and avoiding collisions.

2.5 Overall Formulation of the Optimal Control Problem

To conclude the discussion from 2.1-2.4, the entire OCP problem is described as follows, including the objective function and all constraints:

Objective function: $F = \int_0^{tf} \left(a(t)^2 + \omega(t)^2 \right) dt$

Constraints: 1. $\frac{dx(t)}{dt} = v(t) \bullet cos\theta(t)$

1. $\frac{dy(t)}{dt} = v(t) \bullet sin\theta(t)$

2. $\frac{dv(t)}{dt} = a(t)$

3. $\frac{d\phi(t)}{dt} = \omega(t)$

4. $\frac{d\theta(t)}{dt} = \frac{v(t) \cdot tan\phi(t)}{L}$

5. $v_{min} \leqslant v \leqslant v_{max}$, where $v_{min} = 0$ m/s, $v_{max} = 6$ m/s.

6. $a_{min} \leqslant a \leqslant a_{max}$, where $a_{min} = -2 \ \ m/s^2$, $a_{max} = 2 \ \ m/s^2$.

7. $\phi_{min} \leqslant \phi \leqslant \phi_{max}$, where $\phi_{min} = -\frac{\pi}{4}$, $\phi_{max} = \frac{\pi}{4}$.

8. $w_{min} \leqslant w \leqslant w_{max}$, where $w_{min} = -0.5$ rad/s, $w_{max} = 0.5$ rad/s.

9. $\sqrt{(x - x_{obs})^2 + (y - y_{obs})^2} \geqslant r_{obs} + r_{vehicle}$

By incorporating the vehicle kinematic model, boundary conditions, collision avoidance requirements and objective function, we can finally find an efficient and safe trajectory through the iteration method. However, the OCP needs to be solved through the numerical Solution method.

3. Numerical Solution Method for Single-Vehicle Planning Based on Optimal Control Problem

According to section 2, the trajectory planning problem is formulated as an Optimal Control Problem (OCP). However, the OCP requires numerical techniques to solve. From this section, the basic method and its specific implementation will be introduced. After that, the OCP is transformed into a Nonlinear Programming (NLP) problem.

3.1 Classification of Numerical Solution Methods

Two numerical solution methods for solving Optimal Control Problems (OCPs):

Table 4.Two numerical solution methods for solving OCPs.

Method name	Feature	Key step	Advantages & reasons
Partial Discretization (Indirect Methods)	Only control variables are discretized, while state variables are computed through integration of the system dynamics.	Use variational calculus or the Pontryagin Maximum Principle (PMP) to construct the optimality requirements of the OCP.	This approach is slower than full discretization method, because it requires repeated integration for the state equations in each iteration.
Full Discretization (Direct Methods)	The state and control variables are fully discretized across the whole-time range.	The continuous-time OCP is directly transformed into a Nonlinear. Programming (NLP) problem by discretizing the differential equations of the system dynamics, the control variables, and the state variables. It is then solved using standard NLP solvers.	The simultaneous optimization simplifies the handling of constraints, making it suitable for complex problems. As a result, direct methods are frequently chosen in real-world applications due to their flexibility and robustness.

3.2 Choice of Full Discretization Method

Because of its robustness and flexibility in handling constraints, the full discretization method is employed in our research. The specific reasons for this choice are as follows:

1. The full discretization method allows for the existence of complex constraints, such as collision avoidance. However, these constraints are critical in our experiment, especially in the multi-obstacle environment.
2. Through this method, the whole issue is discretized. Therefore, the possibility of numerical instability can be eliminated while addressing boundary value problems.
3. The Full discretization method transforms the OCP into an NLP. After that, we use a well-established NLP solver to effectively deal with these optimization problems.

3.3 Solution of the NLP Problem

For this problem, we discretized all state and control variables using 200 points to represent the trajectory in terms of the variables $x, y, v, \theta, \phi, a,$ and ω. The resulting NLP problem is formulated as follows:(N=200)

Objective function: $F = \sum_{i=0}^{N-1} (a_i^2 + \omega_i^2) \cdot ?t$

Constraints: 1. $x_{i+1} = x_i + ?t \cdot v_i \cdot cos\theta_i$

1. $y_{i+1} = y_i + ?t \cdot v_i \cdot sin\theta_i$

2. $v_{i+1} = v_i + ?t \bullet a_i$
3. $\phi_{i+1} = \phi_i + ?t \bullet \omega_i$
4. $\theta_{i+1} = \theta_i + ?t \bullet \frac{v_i \bullet tan\phi_i}{L}$
5. $v_{min} \leqslant v \leqslant v_{max}$, for $i = 0,1,...,N$
6. $a_{min} \leqslant a \leqslant a_{max}$, for $i = 0,1,...,N$
7. $\phi_{min} \leqslant \phi \leqslant \phi_{max}$, for $i = 0,1,...,N$
8. $w_{min} \leqslant w \leqslant w_{max}$, for $i = 0,1,...,N$
9. $\sqrt{(x - x_{obs})^2 + (y - y_{obs})^2} \geqslant r_{obs} + r_{vehicle}$, for $i = 0,1,...,N$

Summary

The subsections(3.1-3.3) outline how the single-vehicle trajectory planning problem is transformed into an NLP problem through the discretization process. By applying the full discretization method and NLP solver, we effectively manage complex constraints and can compute the optimal collision-free trajectory in the multi-obstacle environment.

4. Experiments

The following experiments are based on the CasADi framework. This section shows the experimental setup and results of our trajectory planning issue in both single and multiple obstacle situations.

4.1 Single Obstacle Scenario

Parameters:

Table 5. Parameters.

Sign	The parameter name	Value
$r_{vehicle}$	The radius of the car (approximately a circle)	0.5 m
r_{obs}	Radius of the circular obstacle	1 m
(x_{obs}, y_{obs})	Coordinate of the circular obstacle center	(4,5)
(x_0, y_0)	Initial Position	(0,0)
v_0	Initial Speed	3 m/s
θ_0	Initial Orientation angle	0
(x_{Nfe-1}, y_{Nfe-1})	Terminal Position	(15,5)
v_{Nfe-1}	Terminal Speed	3 m/s
θ_{Nfe-1}	Terminal Orientation angle	0
tf	Time Horizon	30 s
	CasADi Version	3.6.6
	Python Version	Python 3.11.4
	Computer model	Legion Y9000p2021H
	CPU	11th Gen Intel(R) Core(TM) i7-11800H @ 2.30GHz 2.30 GHz

Key steps:

1. After determining the parameters, the initial trajectory is generated using linear interpolation between the start and end points, which is independent of the obstacle's position. This leads to a straight-line initial trajectory.
2. The CasADi solver was then used to optimize this initial trajectory. It will dynamically adjust the path to avoid the obstacle while meeting all the constraints.

Results and discussion

In the single-obstacle scenario, the initial trajectory (shown as a dashed blue line) was a linear interpolation from the initial position to the terminal position, regardless of the position of the obstacle. The optimization algorithm successfully generates a collision-free trajectory(shown as a solid line), enabling the vehicle to effectively avoid the obstacle (represented by the circular area in Figure 1).

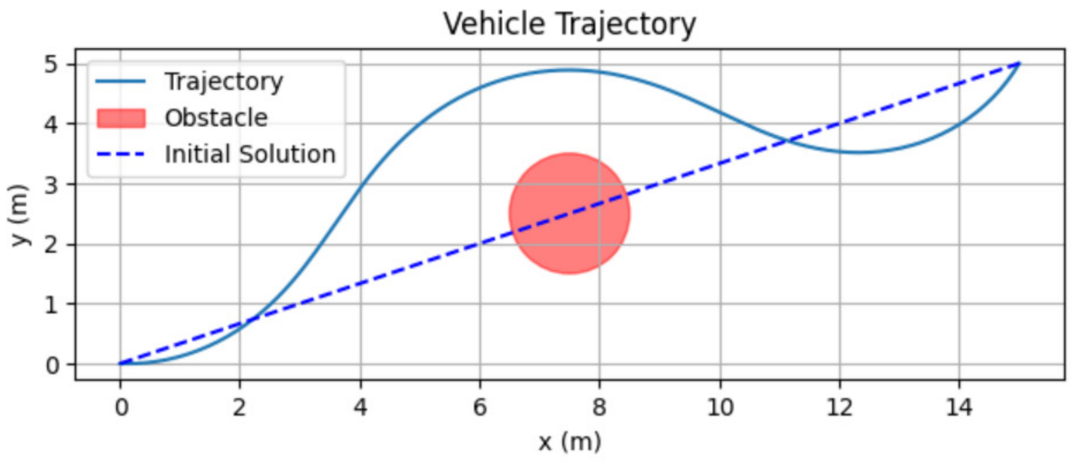


Figure 1. The results in a single-obstacle situation.

```
Total seconds in IPOPT                                   = 2.047
```

Figure 2. Total time consumed in the single-obstacle situation.

This result verifies the algorithm's ability to avoid obstacles in a simple environment. It works by dynamically adjusting the vehicle's trajectory to prevent collisions. The result indicates that this optimization method is effective in solving basic obstacle avoidance problems. Meanwhile, the total time is 2.047 seconds, showing that our model has high efficiency.

4.2 Multiple Obstacle Scenario

Table 6. Parameters.

Sign	The parameter name	Value
$r_{vehicle}$	The radius of the car (approximately a circle)	0.5 m
r_{obs1}	The radius of the circular obstacle	1 m
(x_{obs1}, y_{obs1})	Coordinate of the circular obstacle center	(4,5)
r_{obs2}	The radius of the circumscribed of the square obstacle	1 m
(x_{obs2}, y_{obs2})	The coordinate of the circumscribed circle center of the square obstacle	(8, 2)
r_{obs3}	The radius of the circumscribed of the triangular obstacle	1 m
(x_{obs3}, y_{obs3})	The coordinate of the circumscribed circle centerof the triangular obstacle	(12, 4)
(x_0, y_0)	Initial Position	(0,0)
v_0	Initial Speed	3 m/s
θ_0	Initial Orientation angle	0
(x_{Nfe-1}, y_{Nfe-1})	Terminal Position	(15,5)
v_{Nfe-1}	Terminal Speed	3 m/s
θ_{Nfe-1}	Terminal Orientation angle	$\frac{\pi}{6}$
tf	Time Horizon	30 s

Key steps:
1. For multiple obstacle scenarios, a more complex initial guess was used. We manually selected 20 points to create an initial path to avoid all obstacles. These points were then interpolated to provide a smoother initial trajectory that served as the input to the optimization procedure.
2. After using the CasADi solver, the initial trajectory was improved to a smooth curve, ensuring that the vehicle was able to optimally reach the terminal position while avoiding all obstacles.
Results and discussion
In the multi-obstacle scenario depicted in Figure [2], the optimized trajectory successfully goes through three different obstacles: a circular obstacle (Obstacle 1), a square obstacle (Obstacle 2), and a triangular obstacle (Obstacle 3). Each obstacle is represented by its respective shape, while its circumscribed circle is used to simplify the collision detection.
The initial trajectory (shown as a dashed orange line) was a linear interpolation that roughly avoided the obstacles but did not provide the optimal path. After optimization, the final trajectory (shown as a solid blue line) achieves a smooth trajectory that meets all constraints. It avoids possible collisions with obstacles while maintaining the shortest route distance.

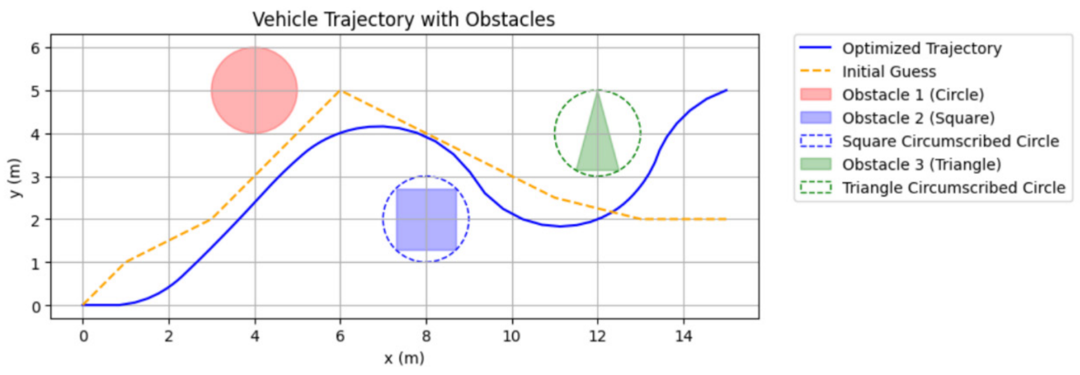
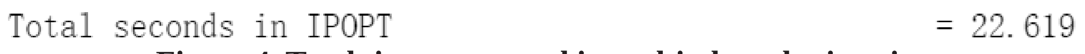


Figure 3. The result in a multi-obstacle situation.



```
Total seconds in IPOPT                                    = 22.619
```

Figure 4. Total time consumed in multi-obstacle situation.

The result demonstrates the algorithm's capability to effectively handle complex environments with multiple obstacle shapes. The optimized path shows smooth curvature tuning around each obstacle. This shows the robustness of this optimization method in multi-obstacle situation. Moreover, the total time is 22.619 seconds, indicating that our algorithm has high efficiency and performance.

4.3 Situations with different parameter values

We then change the initial orientation angle while keeping other conditions constant. The results are shown below.
Case 1:

$$\theta_0 = \frac{\pi}{6}$$

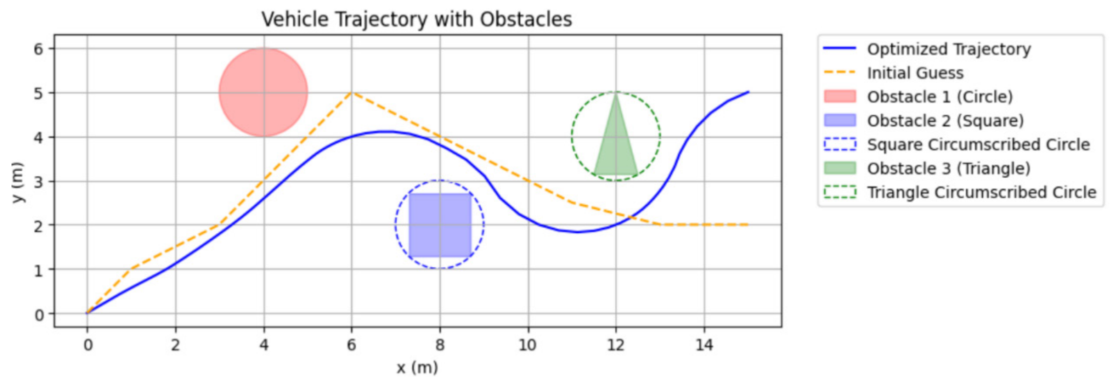


Figure 5. The results with an orientation angle is 30°.

Case 2:

$\theta_0 = \frac{\pi}{3}$

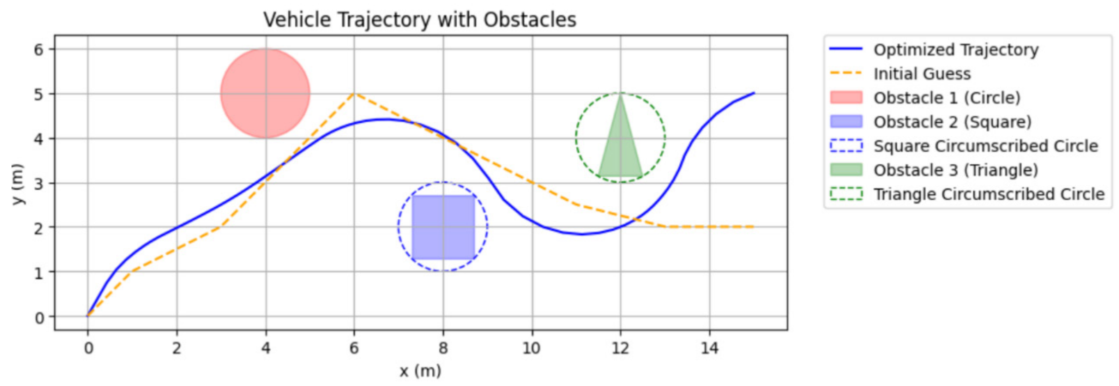


Figure 6. The results with an orientation angle is 60°.

We also alter the initial position and the terminal position of the trajectory.($\theta_0 = 0$, $\theta_{Nfe-1} = \frac{\pi}{6}$). The results are shown in Figures 7, 8.

Case 1:

$(x_0, y_0) = (1,3)$

$(x_{Nfe-1}, y_{Nfe-1}) = (15,4)$

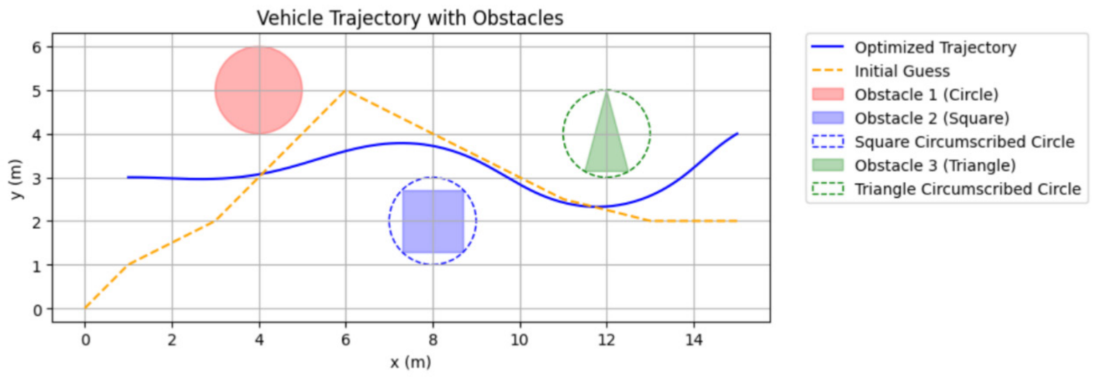


Figure 7. The new results with new start and end positions.

Case 2:

$(x_0, y_0) = (0,5)$

$(x_{Nfe-1}, y_{Nfe-1}) = (15,3)$

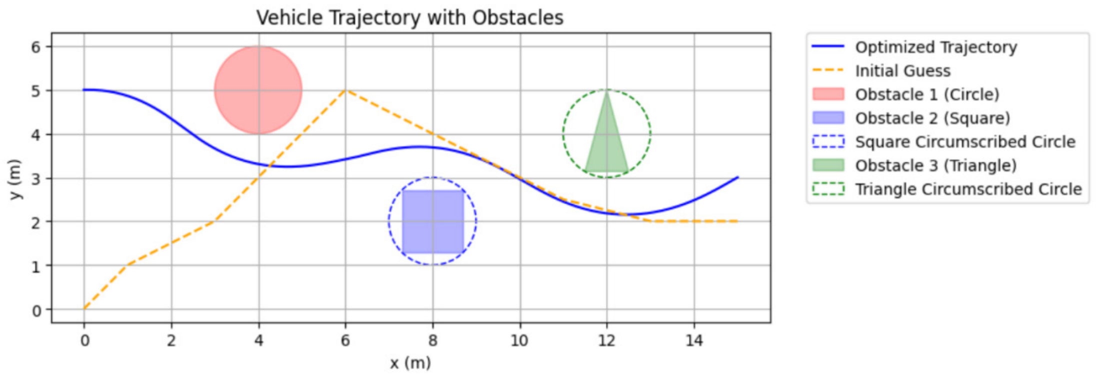


Figure 8. The results in a multi-obstacle situation with new start and end positions.

The results above illustrate that our algorithm can effectively achieve the desired objectives under various param-

eter values.
In summary, the experimental setup, parameters, results,and discussions of the trajectory planning problem in different obstacle situations are described in 4.1 and 4.2. In addition, 4.3 shows the trajectory results across different parameter values. Therefore, our method is not only effective for both single and multiple obstacle scenarios, but robust and adaptable in different constraints.

5. Conclusion

In this work, we first convert the trajectory planning problem into an OCP, then transform it into an NLP through the discretization method. With varying initial and terminal conditions, the proposed approach successfully accomplished the objective well in both single and multiple obstacle scenarios. Our model has a good performance since the results are not dependent on the initial guess and don't take a long time to compute.

6. Reference

[1] Rios-Torres J, Malikopoulos A A. A survey on the coordination of connected and automated vehicles at intersections and merging at highway on-ramps[J]. IEEE Transactions on Intelligent Transportation Systems, 2016, 18(5): 1066-1077.
[2] Yesilyurt A Y, Tunc I, Soylemez M T. A Reservation Method for Multi-agent System Intersection Management with Energy Consumption Considerations[J]. IFAC-PapersOnLine, 2021, 54(2): 246-251.
[3] Masoud Bashiri, & Cody H. Fleming. A Platoon-Based Intersection Management System for Autonomous Vehicles. 2017 IEEE Intelligent Vehicles Symposium (IV) June 11-14, 2017, Redondo Beach, CA, USA, 2017, p.667–672.
[4] Hausknecht M, Au T C, Stone P. Autonomous intersection management: Multi-intersection optimization[C]//2011 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, 2011: 4581-4586.
[5] Li B, Zhang Y, Jia N, et al. Autonomous intersection management over continuous space: A microscopic and precise solution via computational optimal control[J]. IFAC-PapersOnLine, 2020, 53(2): 17071-17076.
[6] Troullinos D, Chalkiadakis G, Papamichail I, et al. Collaborative multiagent decision making for lane-free autonomous driving[C]//Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems. 2021: 1335-1343.
[7] Li B, Zhang Y. Fault-tolerant cooperative motion planning of connected and automated vehicles at a signal-free and lane-free intersection[J]. IFAC-PapersOnLine, 2018, 51(24): 60-67.
[8] Müller E R, Carlson R C, Junior W K. Intersection control for automated vehicles with MILP[J]. IFAC-PapersOnLine, 2016, 49(3): 37-42.
[9] Haoran Wei, Lena Mashayekhy, & Jake Papineau. Intersection Management for Connected Autonomous Vehicles: A Game Theoretic Framework. 2018 21st International Conference on Intelligent Transportation Systems (ITSC) Maui, Hawaii, USA, November 4-7, 2018, p.583–588.
[10]Bai Li, Youmin Zhang, Tankut Acarman, Yakun Ouyang, Cagdas Yaman, and Yaonan Wang.Lane-free Autonomous Intersection Management: A Batchprocessing Framework Integrating Reservation-based and Planning-based Methods.2021 IEEE International Conference on Robotics and Automation (ICRA 2021)Xi'an, China, May 31 - June 4, 2021, p.7915-7921.