Solving parking trajectory planning problems through numerical optimal control

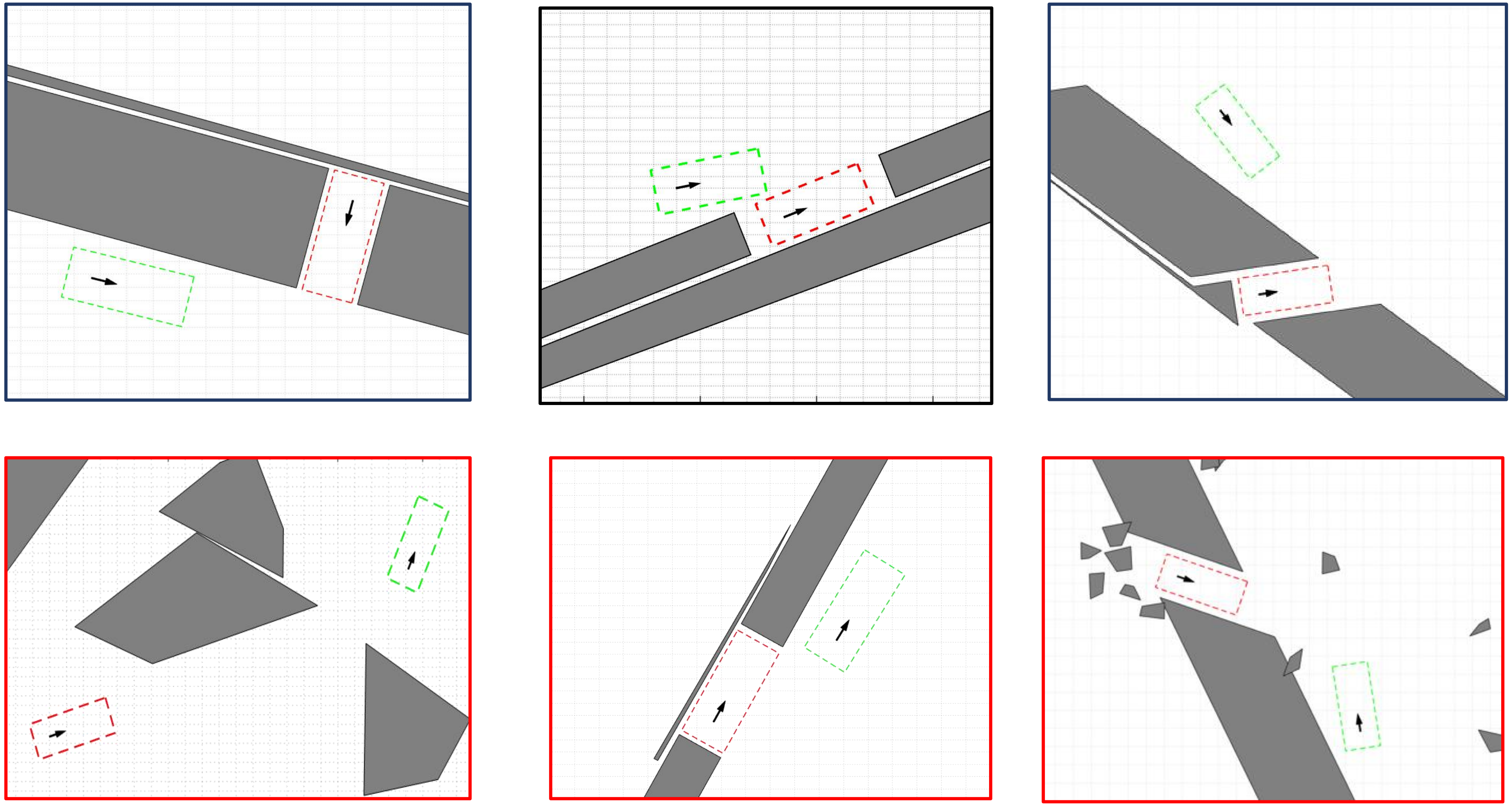
Xiaoming Chen

2023.8.8

CONTENTS

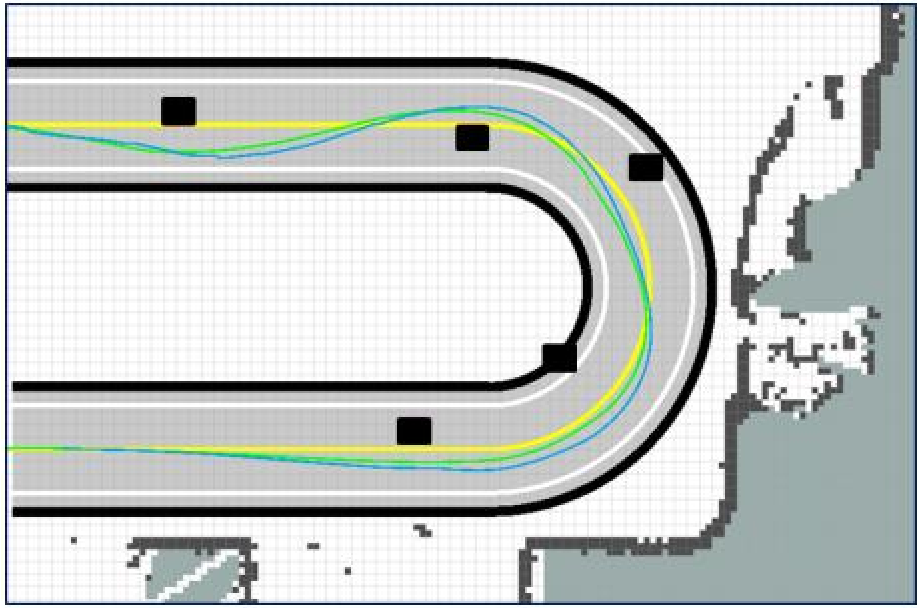
Problem Statement

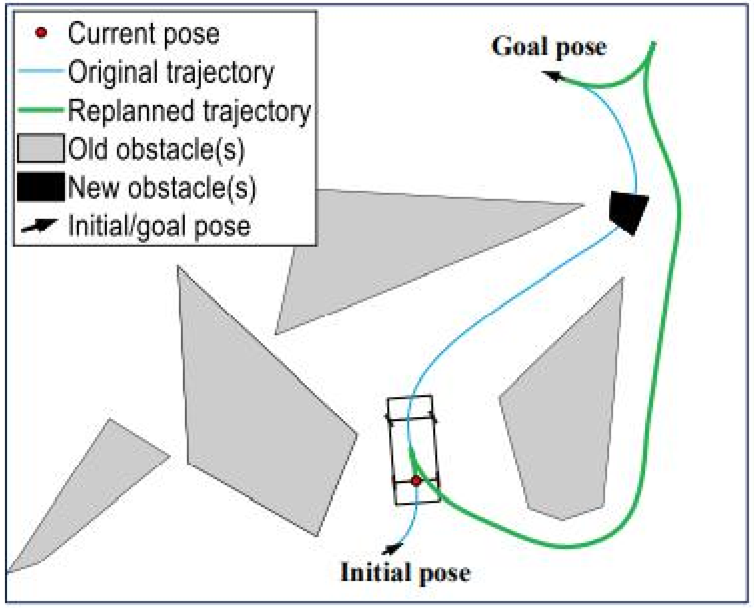
Benchmarks



Problem Statement

Difference

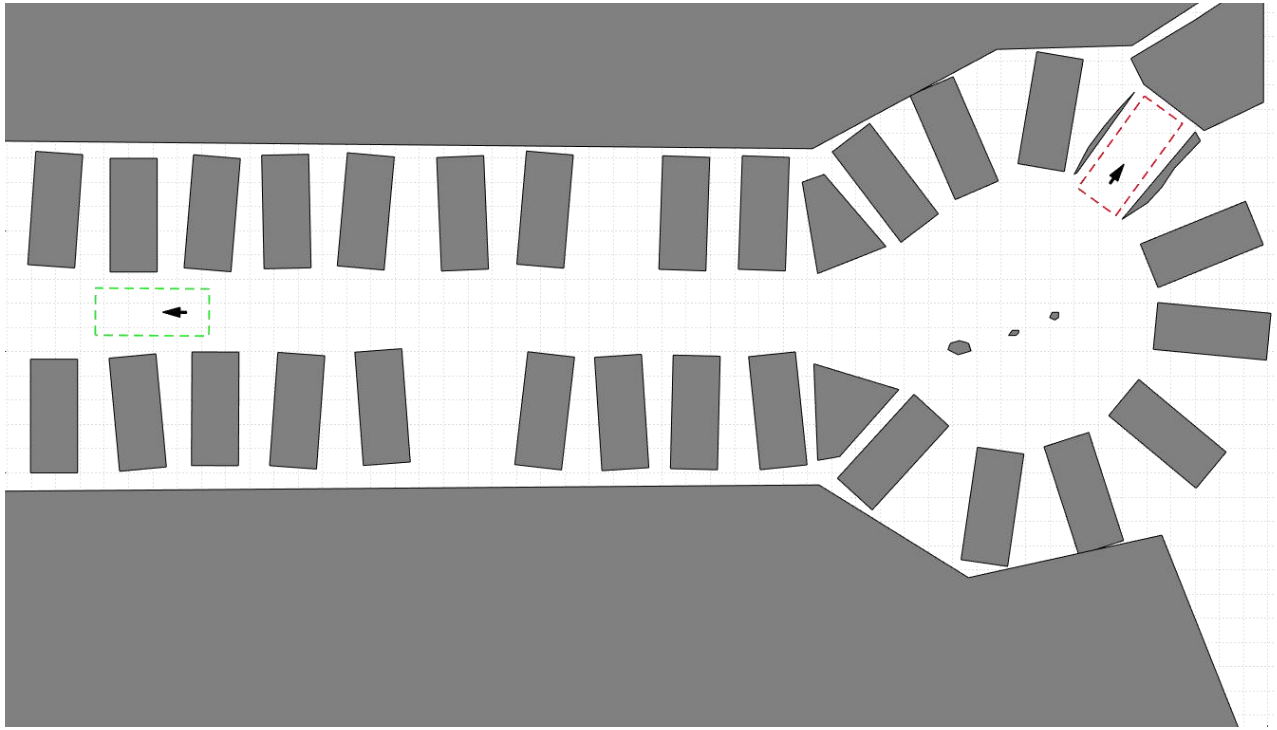




- Reference line
- Low speed
- Direction change → Discontinuous curvature
- Complex environments

Problem Statement

A case of a complex environment



- Collision avoidance
  - Non-convex polygons
  - Small obstacles
  - Avoid collision with edges
- Satisfy vehicle kinematics
- Where to change directions

Method

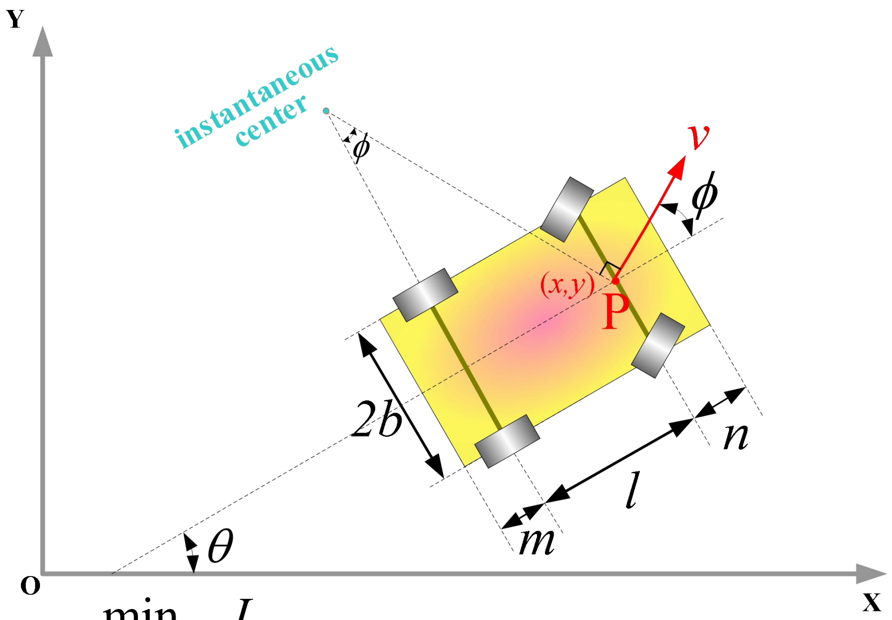
Geometric-based methods

Sampling-and-search-based methods

Optimal control methods

Machine learning methods

Method



$$\min_{z(t),u(t),t_f} J,$$

$$\text{s.t.}, \dot{\mathbf{z}}(t) = f_{\text{KINE}}\left(\mathbf{z}(t), \mathbf{u}(t)\right), \ t \in [0, t_f];$$

$$\mathbf{z}(t) \in [\mathbf{z}_{\min}, \mathbf{z}_{\max}], \ t \in [0, t_f];$$

$$\mathbf{u}(t) \in [\mathbf{u}_{\min}, \mathbf{u}_{\max}], \ t \in [0, t_f];$$

$$\mathbf{z}(0) = \mathbf{z}_{\text{init}}, \ \mathbf{u}(0) = \mathbf{u}_{\text{init}};$$

$$\mathbf{z}(t_f) = \mathbf{z}_{\text{end}}, \ \mathbf{u}(t_f) = \mathbf{u}_{\text{end}};$$

$$fp(\mathbf{z}(t)) \subset \mathrm{C}_{\text{FREE}}, \ t \in [0, t_f].$$

Kinematics constraints

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} x(t) \\ y(t) \\ v(t) \\ \phi(t) \\ \theta(t) \end{bmatrix} = \begin{bmatrix} v(t)\cdot\cos\theta(t) \\ v(t)\cdot\sin\theta(t) \\ a(t) \\ \omega(t) \\ v(t)\cdot\tan\phi(t)/1 \end{bmatrix}, \longrightarrow$$

$$t \in [0, t_f].$$

$$x_{t+1} = x_t + v_t \cos\theta_t \, dt$$

$$y_{t+1} = y_t + v_t \sin\theta_t \, dt$$

$$\theta_{t+1} = \theta_t + v_t \tan\frac{\phi_t}{1} dt$$

$$v_{t+1} = v_t + a_t dt$$

$$\phi_{t+1} = \phi_t + \omega_t dt$$

$$t \in [0, t_f]$$

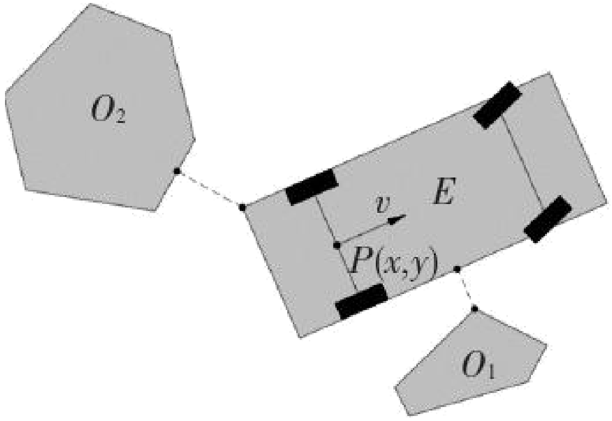
$\mathbf{Z}$: $x(t),\ y(t),\ \theta(t),\ v(t),\ \phi(t)$

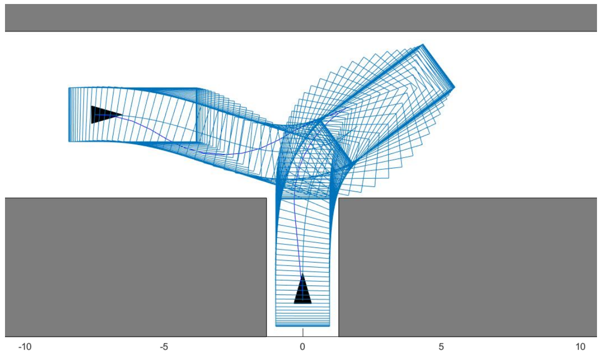
$\mathbf{U}$: $a(t),\ \omega(t)$

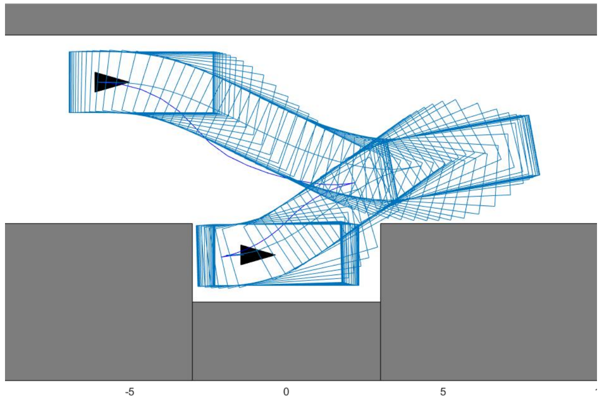
Method

$$
\begin{aligned}
\min_{\mathbf{x},\mathbf{u},\boldsymbol{\lambda}} \quad & \sum_{k=0}^{N} \ell(x_k, u_k) \\
\text{s.t.} \quad & x_0 = x_S, \quad x_{N+1} = x_F \\
& x_{k+1} = f(x_k, u_k), \quad h(x_k, u_k) \leq 0 \\
& (A^{(m)} p_k - b^{(m)})^\top \lambda_k^{(m)} > 0 \\
& \left\| A^{(m)^\top} \lambda_k^{(m)} \right\|_* \leq 1, \quad \lambda_k^{(m)} \succeq_{\mathcal{K}^*} 0 \\
& \text{for } k = 0, \ldots, N, \quad m = 1, \ldots, M
\end{aligned}
$$

$$
\begin{cases}
\boldsymbol{A}x \preccurlyeq_{\mathrm{R}^2} \boldsymbol{b} \\
\boldsymbol{C}_m y \preccurlyeq_{\mathrm{R}^2} \boldsymbol{d}_m \\
\min_{x,y} \|x - y\|_2 > d_{\min}
\end{cases}
$$





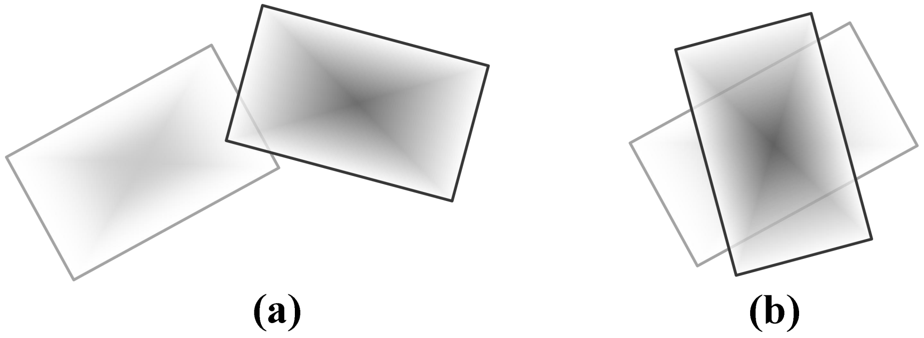


Zhang, Xiaojing, et al. "Autonomous parking using optimization-based collision avoidance." 2018 IEEE Conference on Decision and Control (CDC). IEEE, 2018.

余卓平, 夏浪,and 熊璐."自主泊车路径规划一致性方法." 汽车技术 .08(2018):27-32.

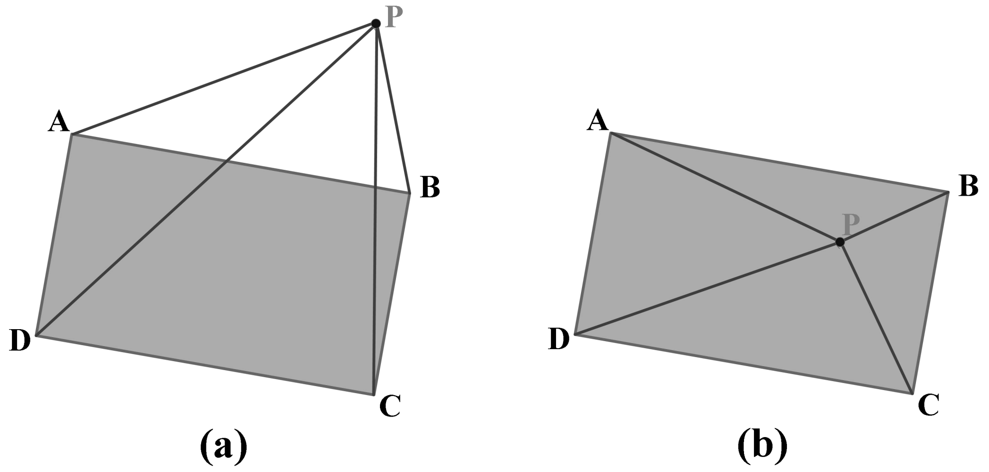
Method

Collision-avoidance Constraints



Two categories of collision

Triangle Area Criterion



$$S_{\Delta PAB} + S_{\Delta PBC} + S_{\Delta PCD} + S_{\Delta PDA} > S_{\square ABCD}$$

Li, Bai, and Zhijiang Shao. "A unified motion planning method for parking an autonomous vehicle in the presence of irregularly placed obstacles." Knowledge-Based Systems 86 (2015): 11-20.

Method

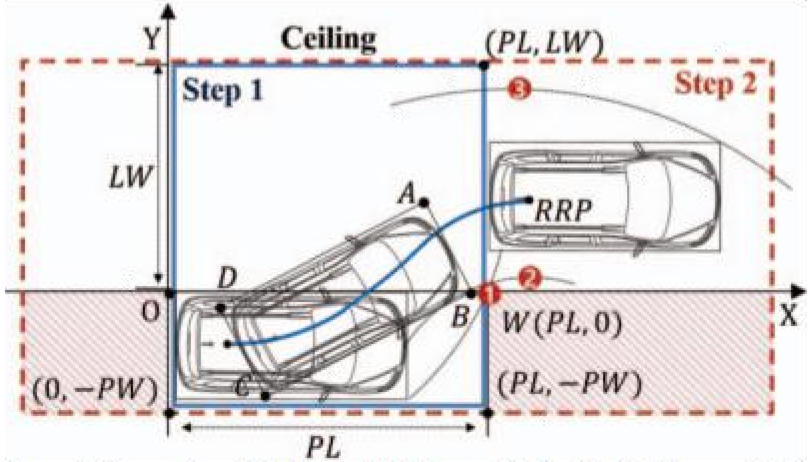
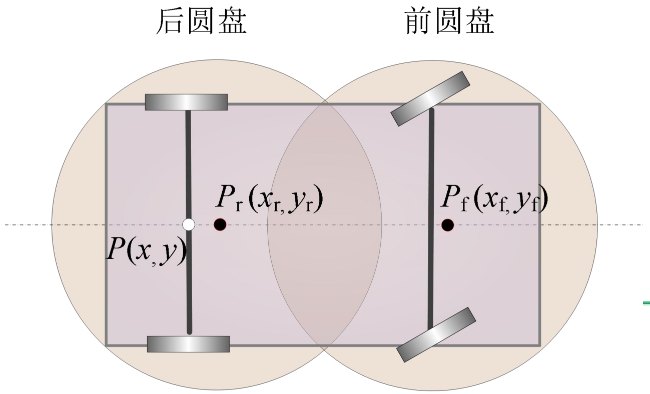
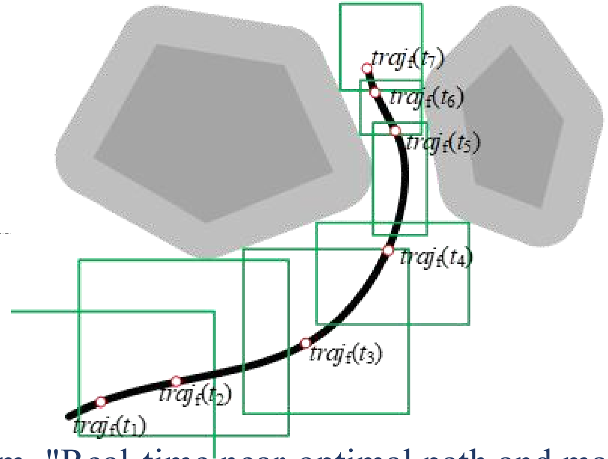
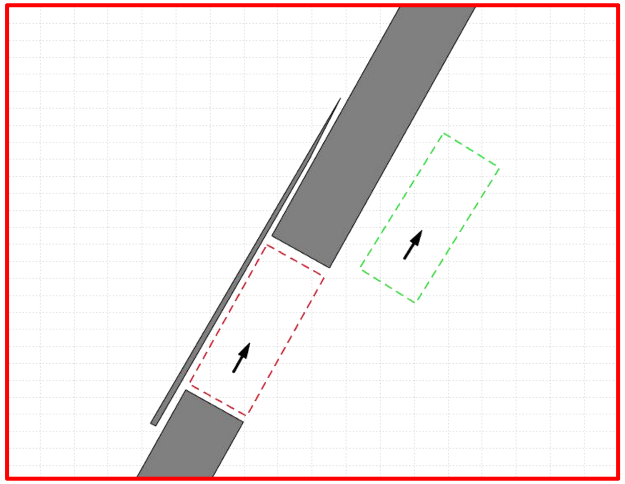
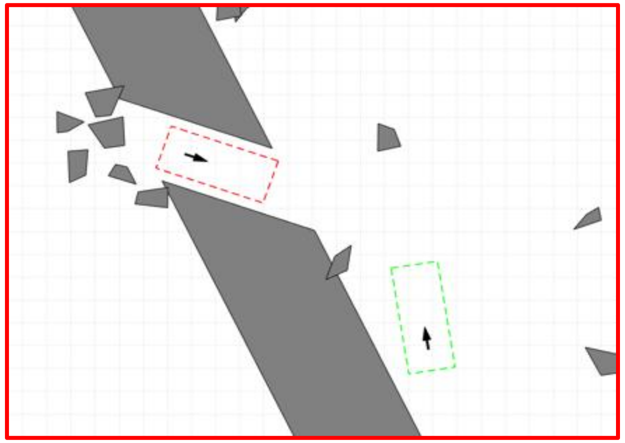


Figure 4. Proposed partitioning method by considering 3-collision possibility (case 1 for Edge B with W, case 2 for Edge C with W and case 3 for edge A with ceiling) during the parallel parking maneuver.





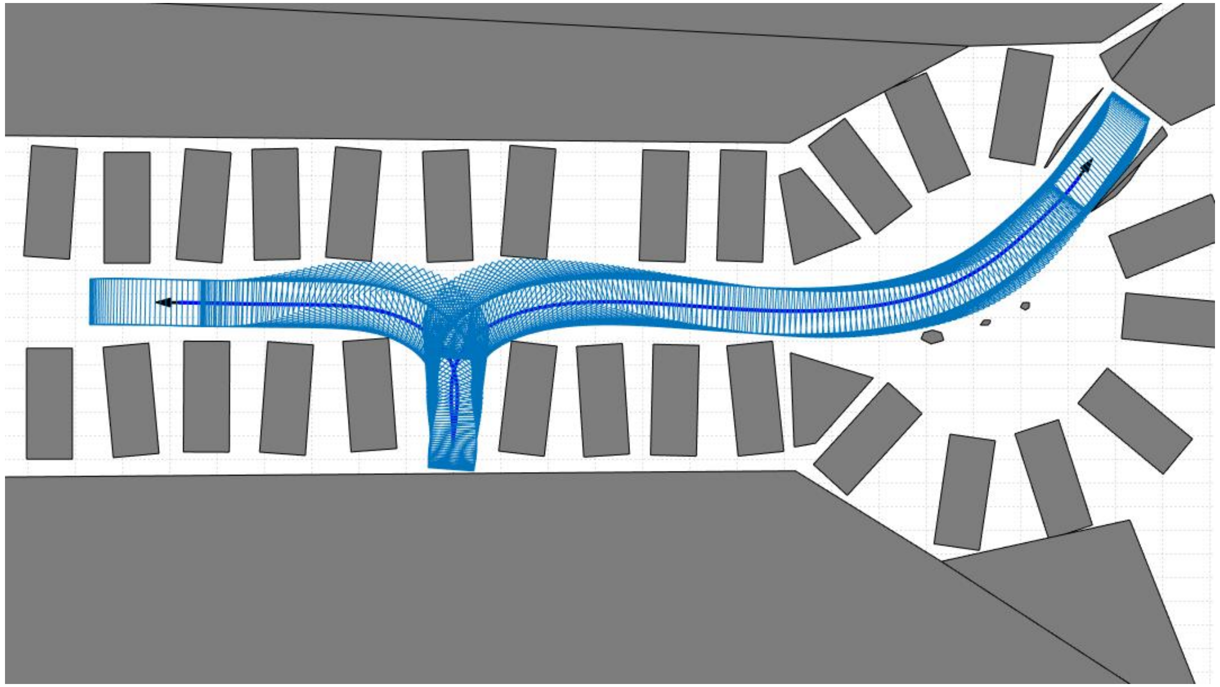




[1] Moon, Jaeyoung, Il Bae, and Shiho Kim. "Real-time near-optimal path and maneuver planning in automatic parking using a simultaneous dynamic optimization approach." 2017 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2017.
[2] Li, Bai, et al. "Optimization-based trajectory planning for autonomous parking with irregularly placed obstacles: A lightweight iterative framework." IEEE Transactions on Intelligent Transportation Systems 23.8 (2021): 11970-11981.

Method

A two-stage method



The final trajectory of the case

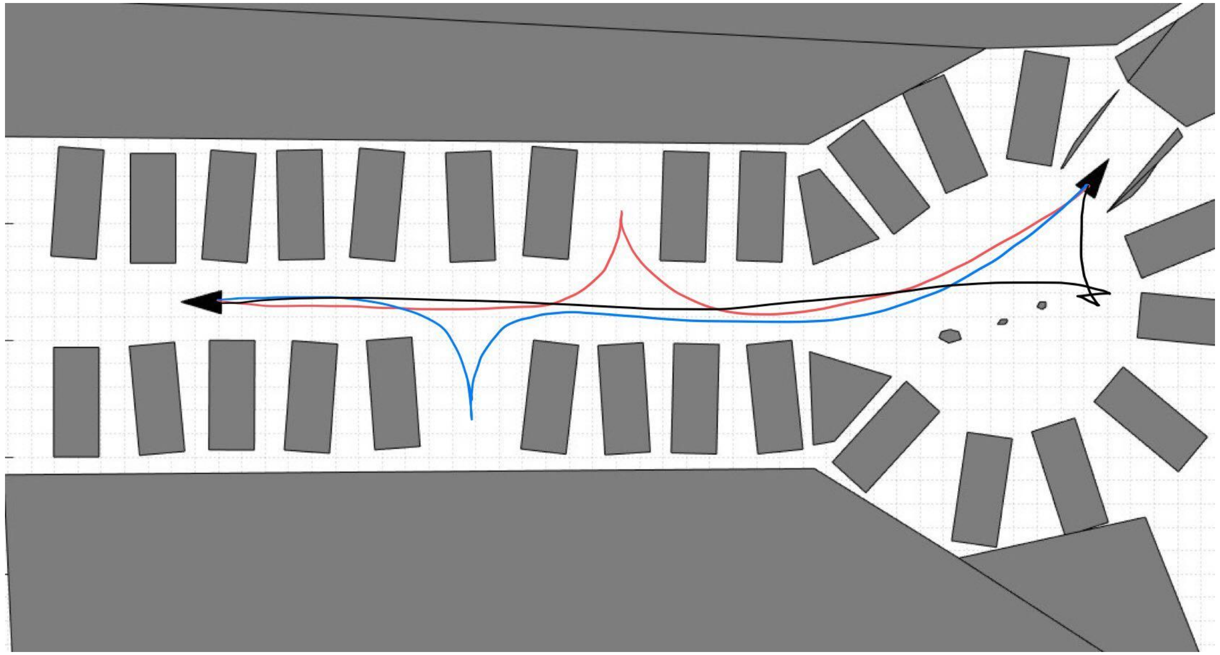
S1
- Get an initial guess via Hybrid A*
- ⬇
- Collision checking
- ⬇
- An intermediate target
- ⬇
- Solve the OCP for two segments
- ⬇
- A feasible solution

⬇

S2 — The final trajectory

Application

Three possible homotopy classes



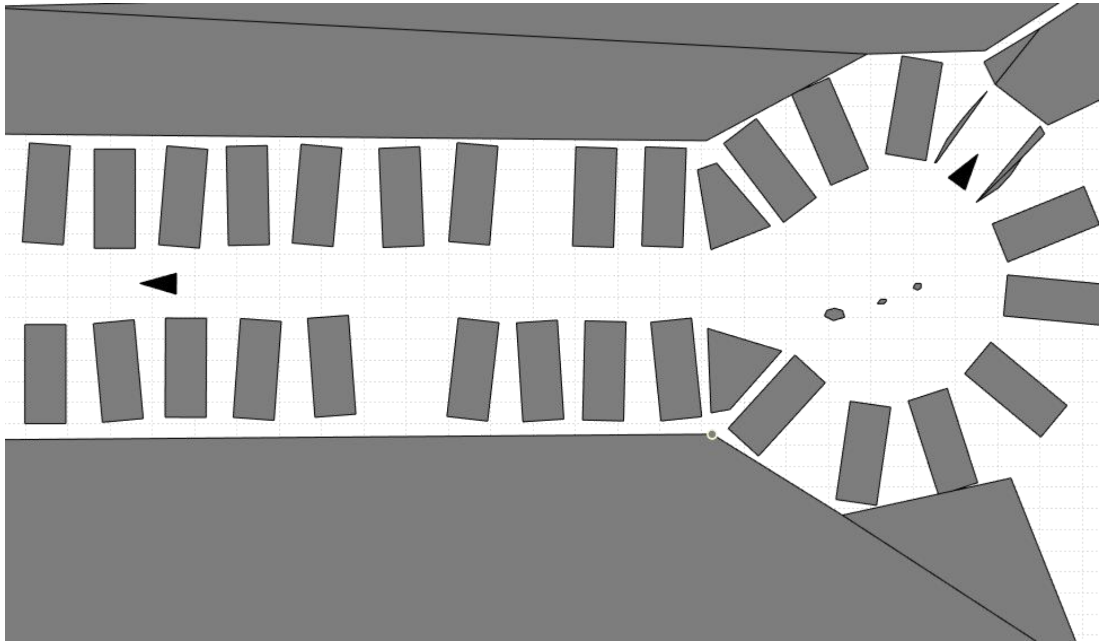
The paths are generated by Hybrid A*(shrink the vehicle size)

How to get different homotopy classes?

- Exchange the start pose and goal pose
- Adjust the resolution of grid map or the size of the environment
- Improve the heuristic function

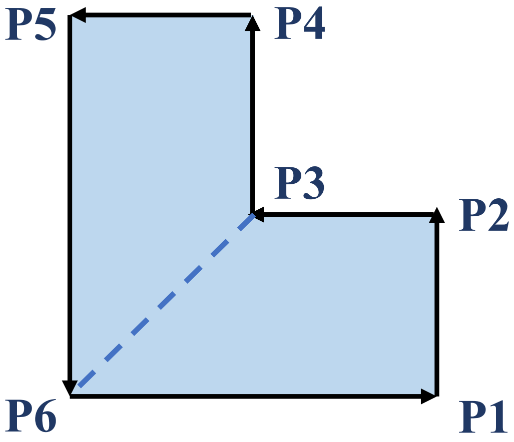
Application

Pre-processed for non-convex polygonal obstacles



Segment non-convex polygon

The vector approach



$$\overrightarrow{P_1P_2} \times \overrightarrow{P_2P_3} > 0$$

$$\overrightarrow{P_2P_3} \times \overrightarrow{P_3P_4} < 0$$

$$\overrightarrow{P_3P_4} \times \overrightarrow{P_5P_6} > 0$$

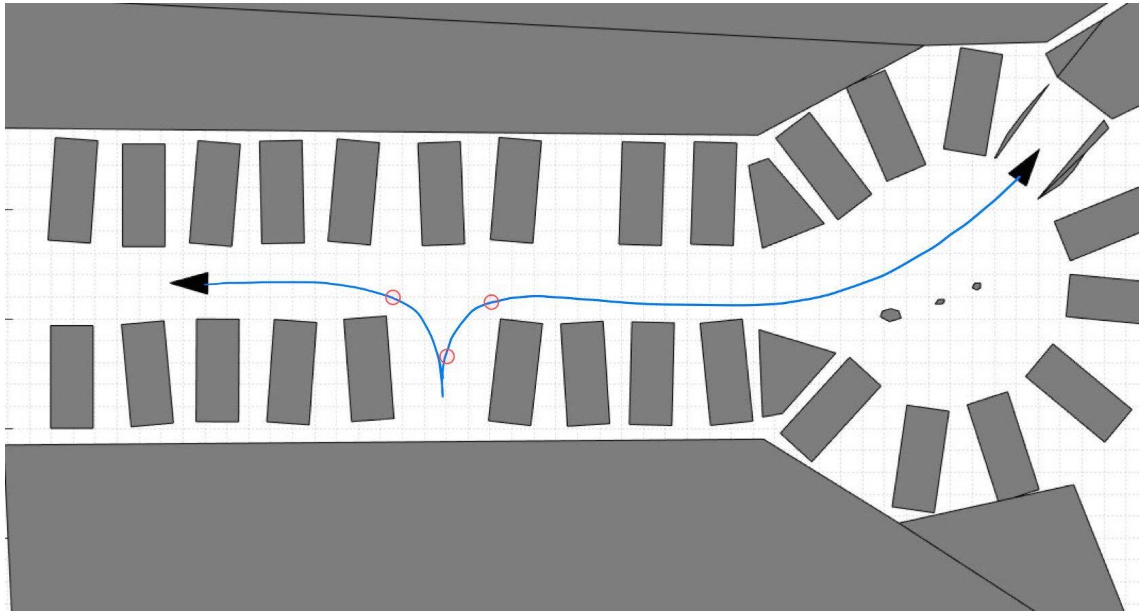
Divide the polygon

$$\overrightarrow{P_5P_6} \times \overrightarrow{P_6P_1} > 0$$

$$\overrightarrow{P_6P_1} \times \overrightarrow{P_1P_2} > 0$$

Application

The first stage



The coarse path may collide at some waypoints

If the coarse path collides

⬇

Choose a collision-free waypoint after the collision point as an intermediate target

⬇

Two segments

⬇

Form the NLP → use the IPOPT

⬇

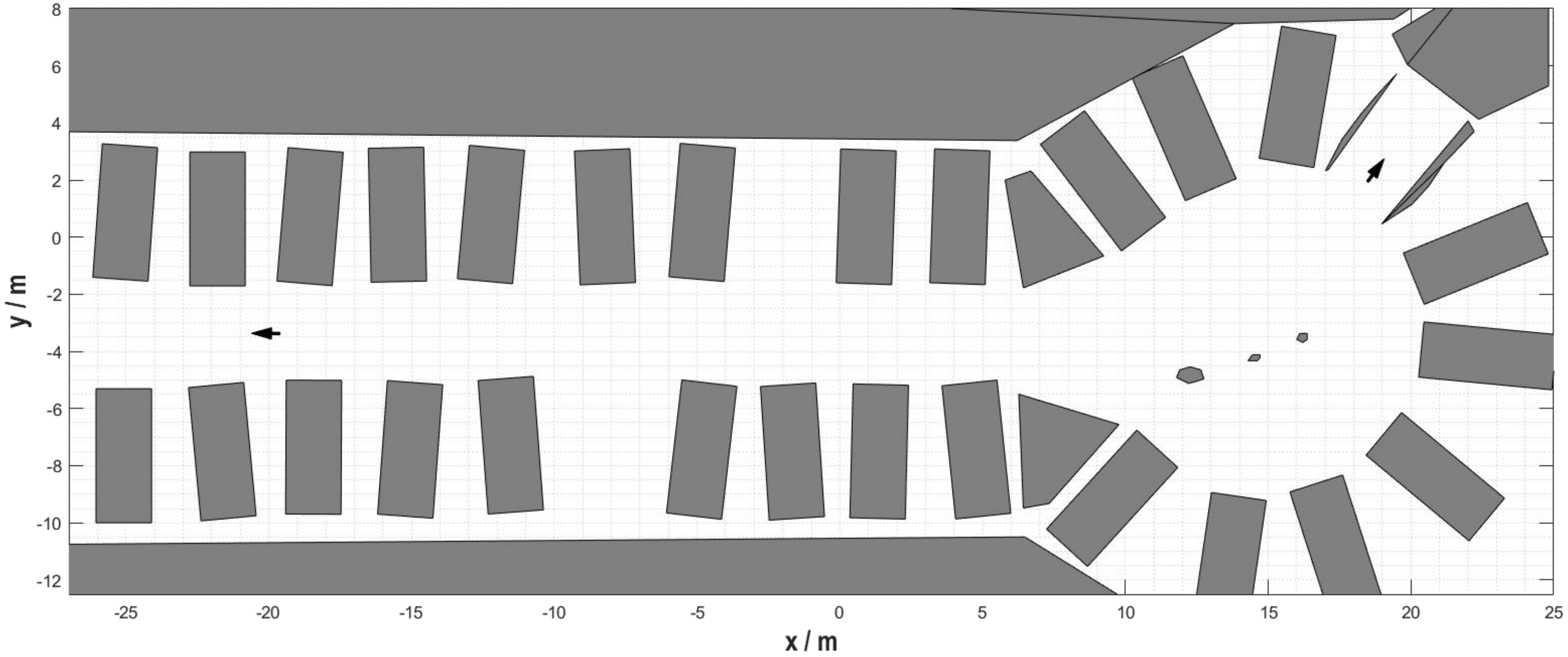
Stitch the optimal solution of two segments

⬇

Obtain a feasible solution

Application

The second stage



The final trajectory of the case

The feasible trajectory

solve the original OCP

The final trajectory

THANK YOU

Xiaoming Chen

2023.8.8