Optimal Parking(KR)

> 궤적 최적화와 RRT* 알고리즘 기반 자율 주차 시스템

🔍 프로젝트 개요

복잡한 주차 환경에서 차량의 물리적 제약조건을 고려하면서 충돌 없는 최적의 주차 경로와 제어 입력을 계산하고 시각화하는 프로젝트입니다.

🔧 개발 환경 및 사용 기술

- 프로그래밍 언어: C++
- 연산 라이브러리: Eigen, OSQP
- 개발 도구: VSCode, Docker
- 개발 환경: Ubuntu:22.04

❓ 목표

최적화 방법을 통한 자율주행 주차 시스템 구축

- 좁은 주차 공간과 다양한 장애물이 존재하는 환경에서 차량의 기구학적 특성을 고려하면서 안전한 주차 궤적을 생성합니다.
- 차량의 비선형적인 동역학 특성과 다양한 제약조건들로 인해 최적화 문제를 해결하기 위해 SQP 방법을 통해 문제를 해결합니다.

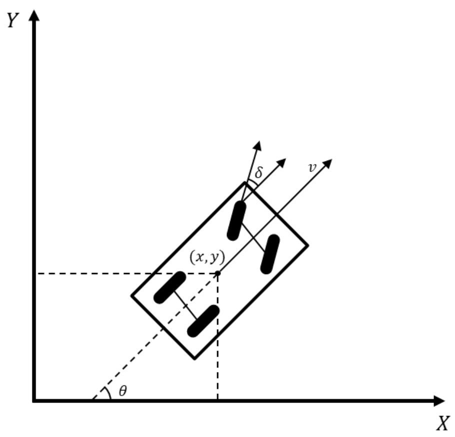
🧮 알고리즘

최적의 주차 궤적을 생성하기 위해 다음과 같은 알고리즘을 구현

- RRT*(Rapidly-exploring Random Tree Star) 알고리즘을 활용하여 장애물 회피 제약조건을 고려한 초기 경로를 생성 .
- 생성된 초기 경로를 기반으로 SQP(Sequential Quadratic Programming) 최적화를 수행.

🧠 시스템 설계

차량의 상태를 표현하고 제어하기 위해 다음과 같은 시스템 모델을 설계



$$\mathbf{x}(t) = \begin{bmatrix} p_x(t) \\ p_y(t) \\ \theta(t) \\ v(t) \\ \delta(t) \end{bmatrix}$$

$$\mathbf{u}(t) = \begin{bmatrix} a(t) \\ \dot{\delta}(t) \end{bmatrix}$$

- 차량의 위치, 방향각, 속력, 조향각을 포함하는 5차원 상태 공간을 정의하여 차량의 움직임을 표현합니다.
- 차량의 가속과 조향각의 변화를 입력으로 사용합니다.

🔢 최적화 문제 수식화 및 QP 변환

1. 최적화 문제 설계

- **목적 함수**

  제어 입력의 크기($u_1^2(t) + u_2^2(t)$)와 목표 지점 및 장애물 회피를 위한 슬랙 변수($s_{\text{goal}}, s_{\text{obs}}(t)$)를 가중치($\lambda_{\text{goal}}, \lambda_{\text{obs}}$)로 조합하여 최소화합니다.

  $J(\mathbf{u}(t), s_{\text{goal}}, s_{\text{obs}}(t)) = \int_0^T \left(u_1^2(t) + u_2^2(t) + \lambda_{\text{goal}} s_{\text{goal}} + \lambda_{\text{obs}} s_{\text{obs}}(t)\right) \mathrm{d}t$

- **제약 조건**

- **동역학**: 상태($\mathbf{x}(t)$)는 동역학 함수 $f(\mathbf{x}(t), \mathbf{u}(t))$를 따릅니다.

  $\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t))$

- **초기 및 최종 조건**:
  - 초기 상태: $\mathbf{x}(0) = \mathbf{x}_{\text{init}}$
  - 최종 상태: $\mathbf{x}(T) = \mathbf{x}_{\text{goal}} + s_{\text{goal}}$
  - 초기 및 최종 제어 입력: $\mathbf{u}(0) = \mathbf{u}_{\text{init}}, \mathbf{u}(T) = \mathbf{u}_{\text{goal}}$

- **장애물 회피**: 상태와 장애물($\mathcal{O}j$) *간의 거리*($d(\mathbf{x}(t), \mathcal{O}j)$)*는 최소 거리*($d_{\min}$) *이상이어야 하며, 슬랙 변수* $s_{\text{obs}}(t) \geq 0$으로 완화됩니다.

  $d(\mathbf{x}(t), \mathcal{O}j) \geq d\min - s_{\text{obs}}(t), \quad s_{\text{obs}}(t) \geq 0, \quad \forall t \in [0, T],\ \forall j \in \{1, \ldots, N_{\text{obs}}\}$

- **물리적 제약**:
  - 속도: $v_{\min} \leq v(t) \leq v_{\max}$
  - 조향각:$\delta_{\min} \leq \delta(t) \leq \delta_{\max}$
  - 가속도: $a_{\min} \leq a(t) \leq a_{\max}$
  - 조향각 속도: $\dot{\delta}_{\min} \leq \dot{\delta}(t) \leq \dot{\delta}_{\max}$

$$
\begin{aligned}
\min_{\mathbf{u}, s_{\text{goal}}, \mathbf{s}_{\text{obs}}} \quad & J(\mathbf{u}(t), s_{\text{goal}}, s_{\text{obs}}(t)) = \int_0^T \left(u_1^2(t) + u_2^2(t) + \lambda_{\text{goal}} s_{\text{goal}} + \lambda_{\text{obs}} s_{\text{obs}}(t)\right) \mathrm{d}t && (1)\\
\text{s.t.} \quad & \dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)), && (2)\\
& \mathbf{x}(0) = \mathbf{x}_{\text{init}}, \quad \mathbf{x}(T) = \mathbf{x}_{\text{goal}} + s_{\text{goal}}, && (3)\\
& \mathbf{u}(0) = \mathbf{u}_{\text{init}}, \quad \mathbf{u}(T) = \mathbf{u}_{\text{goal}}, && (4)\\
& d(\mathbf{x}(t), \mathcal{O}_j) \geq d_{\min} - s_{\text{obs}}(t), \quad s_{\text{obs}}(t) \geq 0, \quad \forall t \in [0, T],\ \forall j \in \{1, \ldots, N_{\text{obs}}\}, && (5)\\
& v_{\min} \leq v(t) \leq v_{\max}, && (6)\\
& \delta_{\min} \leq \delta(t) \leq \delta_{\max}, && (7)\\
& a_{\min} \leq a(t) \leq a_{\max}, && (8)\\
& \dot{\delta}_{\min} \leq \dot{\delta}(t) \leq \dot{\delta}_{\max}, && (9)
\end{aligned}
$$

2. 이산화 및 SQP 변환

연속 시간 문제를 이산화하여 SQP(Sequential Quadratic Programming) 형태로 변환합니다. 이를 통해 비선형 최적화 문제를 반복적으로 선형화된 QP 문제로 근사합니다.

- 이산화된 목적 함수
  시간 구간
  $[0, T]$를 N개의 구간으로 나누고, 각 구간에서 제어 입력과 슬랙 변수의 가중합을 최소화합니다.

  $J = \sum_{k=0}^{N-1} \left(u_1^2[k] + u_2^2[k] + \lambda_{\text{goal}} s_{\text{goal}} + \lambda_{\text{obs}} s_{\text{obs}}[k]\right) \Delta t$

- 이산화된 제약 조건
  - **동역학**: 상태 변화($\Delta \mathbf{x}[k+1]$)는 선형화된 동역학을 따릅니다.

    $\Delta \mathbf{x}[k+1] = \mathbf{A}_k \Delta \mathbf{x}_k + \mathbf{B}_k \Delta \mathbf{u}_k + \mathbf{g}_k, \quad k = 0, 1, \ldots, N-1$

  - **초기 및 최종 조건**:
    - $\mathbf{x}[0] = \mathbf{x}_{\text{init}}, \quad \mathbf{x}[N] = \mathbf{x}_{\text{goal}} + s_{\text{goal}}$
    - $\mathbf{u}[0] = \mathbf{u}_{\text{init}} \quad \mathbf{u}[N] = \mathbf{u}_{\text{goal}}$

  - **장애물 회피**: 거리 제약은 선형화된 형태로 표현됩니다.

$d(\mathbf{x}[k], \mathcal{O}j) + \frac{\partial d(\mathbf{x}[k], \mathcal{O}j)}{\partial \mathbf{x}} \Delta \mathbf{x} \geq d_{\min} - s_{\text{obs}}[k], \quad s_{\text{obs}}[k] \geq 0$

- **물리적 제약**:
  - 속도: $v_{\min} \leq \mathbf{c}_v^T(\mathbf{x}[k] + \Delta\mathbf{x}[k]) \leq v_{\max}$
  - 조향각: $\delta_{\min} \leq \mathbf{c}_\delta^T(\mathbf{x}[k] + \Delta\mathbf{x}[k]) \leq \delta_{\max}$
  - 가속도: $a_{\min} \leq \mathbf{c}_a^T(\mathbf{u}[k] + \Delta\mathbf{u}[k]) \leq a_{\max}$
  - 조향각 속도: $\dot{\delta}_{\min} \leq \mathbf{c}_{\dot{\delta}}^T(\mathbf{u}[k] + \Delta\mathbf{u}[k]) \leq \dot{\delta}_{\max}$

$$
\begin{aligned}
\min_{\Delta\mathbf{u}, s_{\text{goal}}, \mathbf{s}_{\text{obs}}} \quad & J = \sum_{k=0}^{N-1} \left(u_1^2[k] + u_2^2[k] + \lambda_{\text{goal}} s_{\text{goal}} + \lambda_{\text{obs}} s_{\text{obs}}[k] \right) \Delta t && (10) \\
\text{s.t.} \quad & \Delta\mathbf{x}[k+1] = \mathbf{A}_k \Delta\mathbf{x}_k + \mathbf{B}_k \Delta\mathbf{u}_k + \mathbf{g}_k, \quad k = 0, 1, \ldots, N-1, && (11) \\
& \mathbf{x}[0] = \mathbf{x}_{\text{init}}, \quad \mathbf{x}[N] = \mathbf{x}_{\text{goal}} + s_{\text{goal}}, && (12) \\
& \mathbf{u}[0] = \mathbf{u}_{\text{init}}, \quad \mathbf{u}[N] = \mathbf{u}_{\text{goal}}, && (13) \\
& d(\mathbf{x}[k], \mathcal{O}_j) + \frac{\partial d(\mathbf{x}[k], \mathcal{O}_j)}{\partial \mathbf{x}} \Delta\mathbf{x}_k \geq d_{\min} - s_{\text{obs}}[k], \quad s_{\text{obs}}[k] \geq 0, && (14) \\
& v_{\min} \leq \mathbf{c}_v^T(\mathbf{x}[k] + \Delta\mathbf{x}[k]) \leq v_{\max}, && (15) \\
& \delta_{\min} \leq \mathbf{c}_\delta^T(\mathbf{x}[k] + \Delta\mathbf{x}[k]) \leq \delta_{\max}, && (16) \\
& a_{\min} \leq \mathbf{c}_a^T(\mathbf{u}[k] + \Delta\mathbf{u}[k]) \leq a_{\max} && (17) \\
& \dot{\delta}_{\min} \leq \mathbf{c}_{\dot{\delta}}^T(\mathbf{u}[k] + \Delta\mathbf{u}[k]) \leq \dot{\delta}_{\max} && (18)
\end{aligned}
$$

3. QP 문제로 변환

SQP 문제를 일반적인 QP 형태로 변환하여 해결합니다.

- **변수 벡터**

$$
\mathbf{z} = \begin{bmatrix} \Delta\mathbf{x}_1 \\ \Delta\mathbf{x}_2 \\ \vdots \\ \Delta\mathbf{x}_{N+1} \\ \Delta\mathbf{u}_1 \\ \Delta\mathbf{u}_2 \\ \vdots \\ \Delta\mathbf{u}_N \\ s_{\text{goal}} \\ s_{\text{obs},1} \\ s_{\text{obs},2} \\ \vdots \\ s_{\text{obs},N} \end{bmatrix}
$$

- **QP 목적 함수**

  변수 벡터 $\mathbf{z}$(상태 변화 $\Delta\mathbf{x}$, 제어 입력 변화 $\Delta\mathbf{u}$, 슬랙 변수($s_{\text{goal}}, s_{\text{obs}}$)에 대해 이차 형태의 비용을 최소화합니다.

$$
\min_{\mathbf{z}} \quad \frac{1}{2}\mathbf{z}^T \begin{bmatrix} \mathbf{Q} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{R} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{\Lambda_{goal}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{\Lambda_{obs}} \end{bmatrix} \mathbf{z}
$$

  여기서$\mathbf{Q}, \mathbf{R}, \mathbf{\Lambda_{goal}}, \mathbf{\Lambda_{obs}}$는 각각 상태, 제어 입력, 목표 슬랙, 장애물 슬랙에 대한 가중치 행렬입니다.

- **제약 조건**
  - **선형 등식 제약**: 동역학과 초기/최종 조건.

$$
\mathbf{Az} = \mathbf{b}
$$

- **선형 부등식 제약**: 장애물 회피 및 물리적 제약.

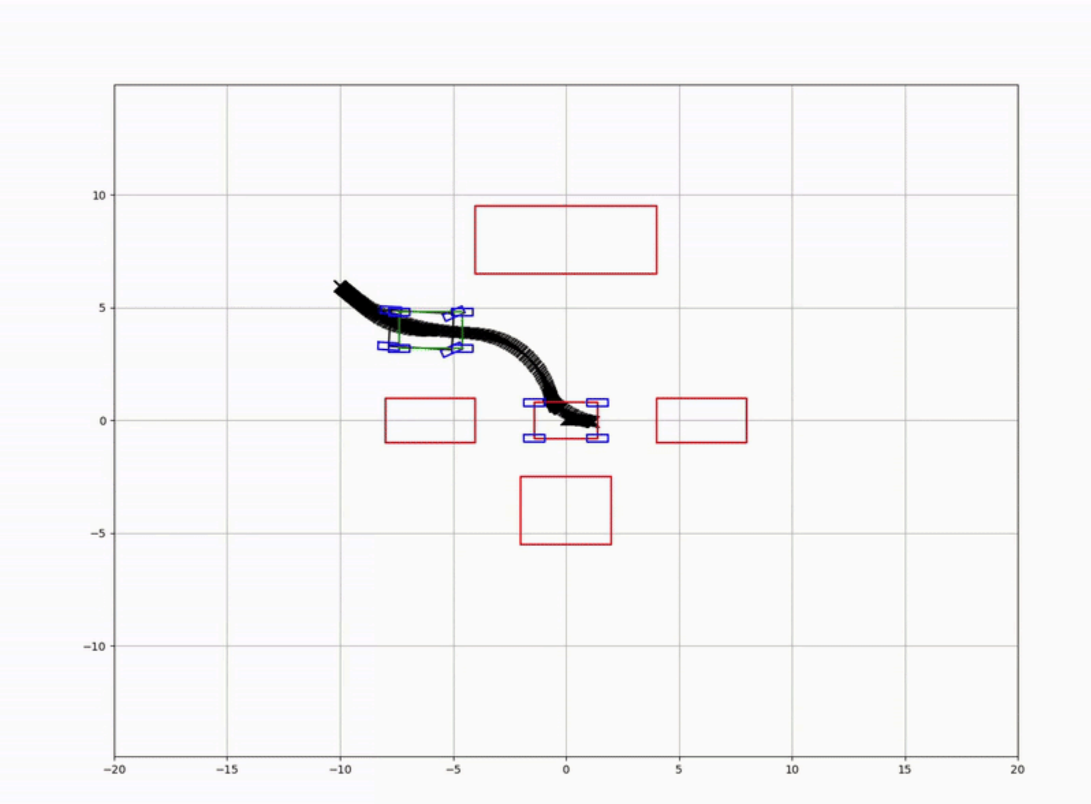
  $$\mathbf{Gz} \leq \mathbf{h}$$

4. 궤적 업데이트

QP 문제를 해결하여 얻은 해($\mathbf{z}$)를 사용하여 상태($\mathbf{x}$)와 제어 입력($\mathbf{u}$)을 업데이트합니다. 이 과정을 반복하여 최적의 궤적을 생성하고, 최종적으로 최적화 문제를 해결합니다.

위 과정을 통해 비선형 최적화 문제를 이산화하고 QP로 변환하여 효율적으로 해결할 수 있습니다.

📊 개발 결과



- 생성된 제어 입력을 통해 시뮬레이션 합니다.
- 다양한 파라미터(yaml) 조정을 통해 서로 다른 주차 상황에 맞게 시스템을 시뮬레이션 가능합니다.