No obstacle optimal control problem

1 Problem Formulation

The problem aims to determine an optimal vehicle path minimizing a given performance index subject to dynamic constraints, corridor constraints, and state-control limits. The mathematical optimization framework is given by:

$$\min_{\mathbf{u},T} \quad J = w_1 T + w_2 \sum_{k=1}^{N} (\delta_k^2 + F_{\mathrm{dr},k}^2) \tag{1}$$

$$\text{subject to} \quad \mathbf{x}_{k+1} = \mathbf{x}_k + \Delta t \cdot f(\mathbf{x}_k, \mathbf{u}_k) \tag{2}$$

$$e_k = -(x_k - x_r)\sin\varphi_r + (y_k - y_r)\cos\varphi_r \in [-\text{margin}, \text{margin}] \tag{3}$$

$$\mathbf{x}_0 = \left[X_0, Y_0, \psi_0, \beta_0, v_0, r_0, s_0\right]^T \tag{4}$$

$$v_0 \geq 0, \quad \beta_0 = 0, \quad r_0 = 0, \quad s_0 = s_{\text{start}} \tag{5}$$

$$\mathbf{x}_{N+1} = \left[X_{\text{end}}, Y_{\text{end}}, \ldots\right]^T, \quad s_{N+1} = s_{\text{end}} \tag{6}$$

$$s_{k+1} \geq s_k, \quad s_k \in [s_{\text{start}}, s_{\text{end}}] \tag{7}$$

$$\delta_k \in [-\delta_{\max}, \delta_{\max}], \quad F_{\mathrm{dr},k} \in [F_{\mathrm{dr,min}}, F_{\mathrm{dr,max}}] \tag{8}$$

$$v_k \geq 0, \quad \beta_k \in [-\beta_{\max}, \beta_{\max}], \quad \psi_k \in [-\psi_{\max}, \psi_{\max}] \tag{9}$$

2 Corridor Constraints

The corridor constraint measures the lateral deviation from the centerline:

$$e_k = -(x_k - x_r)\sin\varphi_r + (y_k - y_r)\cos\varphi_r \tag{10}$$

Here, (x_r, y_r) is the centerline of a given track and φ_r the corresponding orientation, computed in advance by the given track. All the geometry see figure, and the maths derivation here.

3 Segmentation Approach

The entire optimization problem is solved segment-wise for enhanced numerical stability:

1. Divide the entire track into several segments based on arc length:

$$s_{\text{start}}, s_{\text{end}} \quad \text{for each segment} \tag{11}$$

2. Solve each segment independently, propagating the final state as initial conditions for the subsequent segment.

Algorithm 1 Segment-wise Trajectory Optimization Framework

Require: Track data $(X_{\text{ref}}(s), Y_{\text{ref}}(s))$, number of segments N_{seg}
Ensure: Optimal trajectory X_{all}, controls U_{all}, total time T_{all}

Initialize: Set cumulative results $X_{\text{all}} \leftarrow []$, $U_{\text{all}} \leftarrow []$, $T_{\text{all}} \leftarrow 0$
Compute segment boundaries $s_{\text{start}}^{(k)}, s_{\text{end}}^{(k)}$ for each $k = 1, \ldots, N_{\text{seg}}$
Set initial state $X_{\text{start}} \leftarrow [X_0, Y_0, \psi_0, \beta_0, v_0, r_0, s_0]^T$
for $k = 1$ **to** N_{seg} **do**
    Solve: Optimize the sub-problem in $[s_{\text{start}}^{(k)}, s_{\text{end}}^{(k)}]$
        $\min_{X_k, U_k, T_{\text{seg}}} J = w_1 T_{\text{seg}} + w_2 \sum_{i=1}^{N} (\delta_i^2 + F_{\text{dr},i}^2)$
        subject to:
            Vehicle dynamics: $X_{i+1} = X_i + \Delta t f(X_i, U_i)$
            Initial state: $X_0^{(k)} = X_{\text{start}}$
            Final state: $X_{N+1}^{(k)}$ at $s_{\text{end}}^{(k)}$
            Other constraints (corridor, state/control limits, etc.)
    Store optimized states $X_{\text{opt}}^{(k)}$, controls $U_{\text{opt}}^{(k)}$, time $T_{\text{opt}}^{(k)}$
    Update:
        $X_{\text{all}} \leftarrow [X_{\text{all}}, X_{\text{opt}}^{(k)} \setminus X_{\text{start}}]$ ▷ Remove repeated initial point
        $U_{\text{all}} \leftarrow [U_{\text{all}}, U_{\text{opt}}^{(k)}]$
        $T_{\text{all}} \leftarrow T_{\text{all}} + T_{\text{opt}}^{(k)}$
        $X_{\text{start}} \leftarrow X_{\text{opt}}^{(k)}(:, \text{end})$ ▷ Final state becomes next initial state
end for
Return: $X_{\text{all}}, U_{\text{all}}, T_{\text{all}}$

4 Curvature-Based Mesh Refinement

To get better fitting accuracy in hard curves and reduce computational effort on straight lines, interpolation points are adaptively allocated along the trajectory based on the curvature κ, ensuring sufficient resolution in curves.

Let the original 2D trajectory center line be defined as a sequence of discrete points:

$$(x_i, y_i), \quad i = 1, 2, \ldots, M.$$

1. Arc-Length Computation

The cumulative arc-length is computed using Euclidean distance:

$$s_0 = 0, \quad s_i = \sum_{j=1}^{i} \sqrt{(x_j - x_{j-1})^2 + (y_j - y_{j-1})^2}.$$

2. Curvature Estimation

Using the derivative of the tangent angle $\theta(s) = \arctan 2(\frac{dy}{ds}, \frac{dx}{ds})$, the curvature is:

$$\kappa(s) = \left| \frac{d\theta}{ds} \right|.$$

3. Density Weighting Based on Curvature

Let $\kappa_{\text{norm}}(s) = \kappa(s) / \max(\kappa(s))$. Define the density weighting function:

$$w(s) = 1 + a \cdot \kappa_{\text{norm}}(s), \quad a > 0.$$

Where $a > 0$ is the curvature amplification factor, this makes the sampling denser in regions of high curvature.

4. Reweighted Non-equidistant

We begin by defining the reweighted segment lengths based on curvature-aware weights:

$$\Delta\tilde{s}_j = \frac{\Delta s_j}{w_j}, \quad \text{where } \Delta s_j = s_{j+1} - s_j$$

Then, we compute the cumulative weighted arc-length as

$$\tilde{s}_0 = 0, \quad \tilde{s}_i = \sum_{j=1}^{i} \Delta\tilde{s}_j, \quad \text{for } i = 1, 2, \ldots, N.$$

Next, we define an equidistant grid over the reweighted arc-length interval $[0, \tilde{s}_N]$:

$$\tilde{s}_k = k \cdot \Delta\tilde{s}, \quad \text{for } k = 0, 1, \ldots, M,$$

where $\Delta\tilde{s} = \frac{\tilde{s}_N}{M}$ is the desired interpolation interval.

To recover the corresponding points in the original arc-length domain, we define the inverse mapping

$$s_k = \mathcal{S}^{-1}(\tilde{s}_k), \quad \text{where } \mathcal{S}(s) = \int_0^s \frac{1}{w(\xi)}\, d\xi$$

is the cumulative reweighted arc-length function, and $\mathcal{S}^{-1}$ denotes its inverse.

Here, $w(s)$ is a curvature-dependent weighting function that increases sampling density in regions of high curvature.

5. Interpolation of weighted Reference Trajectory

Given the new points s_i, interpolate values of all necessary reference functions:

- Position: $x(s), y(s)$
- Tangent direction: $\phi(s) = \arctan 2\left(\frac{dy}{ds}, \frac{dx}{ds}\right)$
- Derivatives: $\frac{dx}{ds}, \frac{dy}{ds}$

This ensures point density increases in regions with higher curvature $\kappa(s)$, and decreases in flat regions.

5 Initial Guess Based on Local Curvature

Each discretization node is provided with a reasonable physically meaningful initial guess for the velocity and yaw rate to enhance solver convergence. These values are computed based on the local curvature of the given track, using fundamental principles from vehicle dynamics.

5.1 Maximum Speed Estimation

The maximum admissible velocity in a curve is constrained by the frictional force that can be exerted between the tires and the road. Assuming the vehicle follows a circular arc of radius R with lateral acceleration limited by the friction coefficient μ, we impose the constraint:

$$\frac{mv^2}{R} \leq \mu m g, \tag{12}$$

where m is the vehicle mass, g is gravitational acceleration, and v is the vehicle's velocity.

Solving for v yields the maximum velocity under the no-slip condition:

$$v_{\max} = \sqrt{\mu g R}. \tag{13}$$

5.2 Initial Speed Guess

The initial guess for the velocity v_{initial} at each node is then defined by the minimum of the dynamic limit and an external speed limit v_{limit}:

$$v_{\text{initial}} = \min(v_{\max}, v_{\text{limit}}). \tag{14}$$

This ensures that the initial guess is both physically feasible and compliant with safety or operational constraints.

5.3 Curvature and Turning Radius

The radius of curvature R is computed from the inverse of the path curvature κ:

$$R = \frac{1}{\kappa}. \tag{15}$$

5.4 Initial Yaw Rate Guess

Given the velocity estimate and curvature radius, the initial guess for the yaw rate r_{initial} is obtained using the relation:

$$r_{\text{initial}} = \frac{v_{\text{initial}}}{R}. \tag{16}$$

Combining Equations, we can directly express the yaw rate guess in terms of curvature:

$$r_{\text{initial}} = v_{\text{initial}} \cdot \kappa. \tag{17}$$

Summary

Equations (13)–(17) define a curvature-aware initialization strategy for velocity and yaw rate, which reflects the local geometric and physical constraints of the path. This improves the quality of the initial guess and accelerates the convergence of the optimization solver.

6 Optimization Solution Procedure

The optimization is solved using IPOPT, minimizing:

$$J = w_1 T_{\text{seg}} + w_2 \Delta t \sum_{k=1}^{N} (\delta_k^2 + F_{dr,k}^2) \tag{18}$$

The solver iteratively updates control and state trajectories until convergence criteria (tolerances and maximum iterations) are satisfied.

7 Summary

The trajectory optimization integrates:

- A detailed vehicle dynamics model with nonlinear tire characteristics.
- adjust the density of interpolation based on curvature.
- Segment-wise decomposition for efficient numerical solution, to decrease the time complexity.
- Node-specific initialization to help and ensure robust convergence.

This comprehensive mathematical and computational framework effectively addresses the trajectory optimization problem for vehicle dynamics.