Self-Driving Cars and Optimal Trajectory

Pavan Kumar, Danyang Long, Rundong Huang,
Saksham Malhotra, Rahul Radhakrishnan

Supervisor: Prof. Dr. Callies

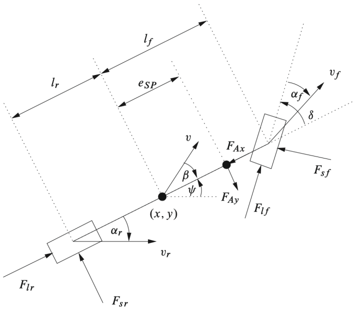
10.02.2023

Overview

- Single track model
- Optimal control problem formulation
- Road geometry construction
- Alternate optimal control problem formulation using Lagrange collocation
- Model predictive control

Single Track model

Single track model



v: velocity

l_r, l_f: distance from COG to rear and front wheels

F_{sr}, F_{sf}: lateral tire forces

F_{lr}, F_{lf}: longitudnal tire forces

F_{Ax}, F_{Ay}: Air drag

α_f, α_r, β: side slip angles

ψ: yaw angle

ω_z: angular velocity of COG

δ: steering angle

ODEs for Single Track model

$$\dot{x} = vcos(\psi - \beta)$$

$$\dot{y} = vsin(\psi - \beta)$$

$$\dot{v} = \frac{1}{m}\Big[(F_{lr} - F_{Ax})cos\beta - (F_{sr} - F_{Ay})sin\beta + F_{lf}cos(\delta + \beta) - F_{sf}sin(\delta + \beta)\Big]$$

$$\dot{\beta} = \omega_z - \frac{1}{mv}\Big[(F_{lr} - F_{Ax})sin\beta + (F_{sr} - F_{Ay})cos\beta + F_{lf}sin(\delta + \beta) + F_{sf}cos(\delta + \beta)\Big]$$

$$\dot{\psi} = \omega_z$$

$$\dot{\omega_z} = \frac{1}{I_{zz}}\Big[F_{sf}.l_f.cos\delta - F_{sr}.l_r + F_{sf}.l_f.sin\delta\Big]$$

$$\dot{\delta} = \omega_\delta$$

Description of single track model forces

The longitudnal tire forces are related to the braking/accelerating forces and the rolling resistances at the tires by

$$F_{lf} = -F_{Bf} - F_{Rf}$$
$$F_{lr} = -F_{Br} - F_{Rr}$$

We assume that the breaking/accelerating force F_B is distributed as

$$F_{Bf} = \begin{cases} \frac{2}{3}F_B, & F_B > \Delta \\ \frac{5}{6}F_B - \frac{1}{4\Delta}F_B^2 + \frac{1}{12\Delta^3}F_B^4, & |F_B| \leq \Delta \\ F_B, & \text{otherwise} \end{cases}$$

$$F_{Br} = \begin{cases} \frac{1}{3}, F_B & F_B > \Delta \\ \frac{2}{3\Delta}F_B^2 - \frac{1}{3\Delta^2}F_B^3, & 0 < F_B \leq \Delta \\ 0, & \text{otherwise} \end{cases}$$

where $\Delta = 0.01$. $F_B > 0$ corresponds to acceleration and $F_B < 0$ corresponds to braking.

Description of single track model forces

Side slip angles at the front and rear wheels are given by,

$$\alpha_f = \delta - arctan\left(\frac{l_f.\dot{\psi} - vsin\beta}{vcos\beta}\right)$$

$$\alpha_r = arctan\left(\frac{l_r.\dot{\psi} + vsin\beta}{vcos\beta}\right)$$

The lateral forces on the tires are given by Pacejka's magic formula [11]

$$F_{sf}(\alpha_f) = D_f.sin(C_f arctan(B_f\alpha_f - E_f(B_f\alpha_f - arctan(B_f\alpha_f))))$$
$$F_{sr}(\alpha_r) = D_r.sin(C_r arctan(B_r\alpha_r - E_r(B_r\alpha_r - arctan(B_r\alpha_r))))$$

Description of single track model forces

The air drag along the lateral direction of the vehicle is assumed to be zero and along the longitudnal direction it is given by

$$F_{Ax} = \frac{1}{2} c_\omega \rho A v^2$$

$$F_{Ay} = 0$$

Optimal Control problem formulation

Optimal Control problem formulation

What is optimal control ?

- To optimize the controls and state trajectories for a dynamical system over a period of time.

How does it work ?

- Optimize the cost function with respect to control variables while satisfying the ODEs, control & state constraints.

Defining variables

State variables

- x: displacement in x direction
- y: displacement in y direction
- v: velocity of car
- δ: steering angle
- ψ: yaw angle
- ω_z: angular velocity of COG

Control Variables

- ω_δ: steering velocity
- F_B: breaking or acceleration force

Defining Variables

State variables -

$$z : t \to \mathbb{R}^7, z(t) = (x, y, v, \beta, \psi, \omega_z, \delta)^T(t)$$

Control Variables -

$$u : t \to \mathbb{R}^2, u(t) = (w_\delta, F_B)^T(t)$$

ODE constraints

dynamics of states z is described by, $f : \mathbb{R} \times \mathbb{R}^7 \times \mathbb{R}^2 \to \mathbb{R}^7$

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

$$:= \begin{bmatrix} v cos(\psi - \beta) \\ v sin(\psi - \beta) \\ \frac{1}{m}\left[(F_{lr} - F_{Ax}) cos\beta - (F_{sr} - F_{Ay}) sin\beta + F_{lf} cos(\delta + \beta) - F_{sf} sin(\delta + \beta)\right] \\ \omega_z - \frac{1}{mv}\left[(F_{lr} - F_{Ax}) sin\beta + (F_{sr} - F_{Ay}) cos\beta + F_{lf} sin(\delta + \beta) + F_{sf} cos(\delta + \beta)\right] \\ \omega_z \\ \frac{1}{I_{zz}}\left[F_{sf}.l_f.cos\delta - F_{sr}.l_r + F_{sf}.l_f.sin\delta\right] \\ \omega_\delta \end{bmatrix}$$

Optimal Control Problem formulation

Objective function : final time

$$t_f$$

Control constraints -:

$$c(t, z(t), u(t)) := \begin{bmatrix} w_\delta(t) - 0.5 \\ -0.5 - w_\delta(t) \\ -5000 - F_B(t) \\ F_B(t) - 15000 \end{bmatrix} \leq 0_{\mathbb{R}^4} \tag{1}$$

state constraints,

$$s(t, z(t)) := \begin{bmatrix} y(t) - y_{up}(x(t)) \\ y_{down}(x(t)) - y(t) \end{bmatrix} \leq 0_{\mathbb{R}^2} \tag{2}$$

where y_{up}, y_{down} are functions describing road boundary y-coordinates

Final and Initial conditions constraints

$$\gamma(z(t_0), z(t_f)) := \begin{bmatrix} x(t_0) \\ y(t_0) \\ v(t_0) \\ \psi(t_0) \\ \omega_z(t_0) \\ \delta(t_0) \\ x(t_f) - x_{END} \end{bmatrix} = 0^{\mathbb{R}^7}$$

- t_0: Initial time
- t_f: Final time

Converting the OCP to a fixed time problem

Assuming $t_0 = 0$, the final time t_f is free so we use the linear time transformation. To convert the OCP to a fixed time problem we use the following transformation [7]

$$t(\tau) = \tau t_f, \tau \in [0, 1] \tag{3}$$

and introduce the state t_f which is constant for $\tau \in [0, 1]$ and
$\dfrac{d}{d\tau} t_f(\tau) = 0$
The states z are transformed by

$$\bar{z}(\tau) := z(\tau t_f)$$
$$\bar{u}(\tau) :== u(\tau t_f)$$

Converting the OCP to a fixed time problem

Then,

$$\begin{aligned}\frac{d}{d\tau}\bar{z}(\tau) &= \dot{z}(t(\tau)).\frac{d}{d\tau}(t(\tau))\\ &= f(t(\tau), z(t(\tau)).,u(t(\tau))).t_f\\ &= t_f f\big(\tau t_f, \bar{z}(\tau), \bar{u}(\tau)\big)\end{aligned}$$

Now we redefine $\bar{z}(\tau) = (\bar{z}(\tau), t_f(\tau))$ to include the new state t_f
Then the optimization problem is given by,

Optimal Control Problem formulation

Minimize

$$t_f$$

with respect to $\bar{z}$ and $\bar{u}$, subject to the ODEs,

$$\dot{\bar{z}}(t) = f\Big(\tau t_f, \bar{z}(t), \bar{u}(t)\Big)$$

and constraints,

$$\bar{c}\Big(\tau t_f, \bar{z}(t), \bar{u}(t)\Big) := c(t(\tau), z(t(\tau)), u(t(\tau))) \leq 0_{\mathbb{R}^6}$$

$$\bar{s}\Big(\tau t_f, \bar{z}(t), \bar{u}(t)\Big) := s(t(\tau), z(t(\tau)), u(t(\tau))) \leq 0_{\mathbb{R}^2}$$

$$\bar{\gamma}(\bar{z}_0, \bar{z}_N) = 0_{\mathbb{R}^7}$$

Full Discretization for states and controls

To include the ODE constraints we use the full discretization method. [7]
We define a grid with N steps,

$$G_N = \{\tau_i | i = 0, 1, 2, 3......N\}$$

We use an ODE method **ode** which given the states at $\bar{z}_j$ computes the states at $\bar{z}_{j+1}$ such that

$$\bar{z}_{j+1} = ode(\bar{z}_j, h), j = 0, 1, \ldots, N-1 \text{ and } h = \frac{1}{N} \tag{4}$$

We discretize the control space with piecewise linear functions on the grid G_N to get $\{m_0, m_1, \ldots, m_{N-1}\}$ controls where $m_i \in \mathbb{R}^2$

OCP formulation using full discretization

Then the discretized optimisation problem is,
Minimize

$$t_f(N)$$

with respect to $(\bar{z}_0, \bar{z}_2, \ldots, \bar{z}_N, m_0, \ldots, m_{N-1}) \in \mathbb{R}^{8N+2(N+1)}$,
subject to,

$$\bar{z}_{j+1} - ode(\bar{z}_j, h) = 0_{\mathbb{R}^8},\ j = 0, 1, \ldots, N-1$$

OCP formulation using full discretization

and constraints,

$$\bar{c}(\tau_j, \bar{z}_j, \bar{u}_M(\tau_j; m)) \leq 0_{\mathbb{R}^6},\ j = 0, 1, \ldots, N-1$$
$$\bar{s}(\tau_j, \bar{z}_j) \leq 0_{\mathbb{R}^2}\ j = 0, 1, \ldots, N-1$$
$$\bar{\gamma}(\bar{z}_0, \bar{z}_N) = 0_{\mathbb{R}^7}$$

Collecting all the variables and constraints into matrices, we get the optimization problem,
Minimise

$$J(\bar{q}) = t_f(\tau_N)$$

with respect to $(\bar{z}_0, \bar{z}_2, \ldots, \bar{z}_N, m_1, \ldots, m_{N-1}) \in \mathbb{R}^{8(N+1)+2N}$,
subject to,

with constraints

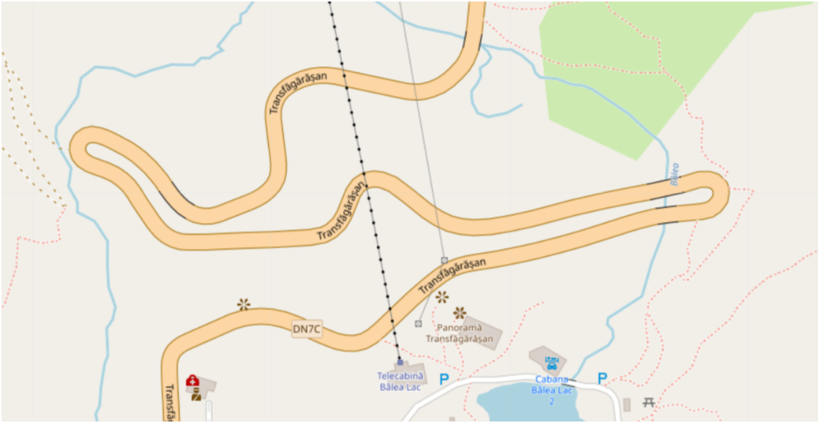
$$G(\bar{q}) = \begin{bmatrix} \bar{c}(\tau_0, \bar{z}_0, \bar{u}_M(\tau_0; m)) \\ \bar{c}(\tau_1, \bar{z}_1, \bar{u}_M(\tau_1; m)) \\ \vdots \\ \bar{c}(\tau_N, \bar{z}_N, \bar{u}_M(\tau_N; m)) \\ \bar{s}(\tau_0, \bar{z}_0) \\ \bar{s}(\tau_1, \bar{z}_N) \\ \vdots \\ \bar{s}(\tau_N, \bar{z}_N) \end{bmatrix} \leq 0_{\mathbb{R}^{8(N+1)+2N}} \tag{5}$$

and,

$$H(\bar{q}) = \begin{bmatrix} \bar{z}_1 - ode(\bar{z}_0, h) \\ \bar{z}_2 - ode(\bar{z}_1, h) \\ \vdots \\ \bar{z}_N - ode(\bar{z}_{N-1}, h) \\ \bar{\gamma}(\bar{z}_0, \bar{z}_N) \end{bmatrix} = 0_{\mathbb{R}^{8(N+1)+9}} \tag{6}$$

Road Geometry Construction

Road Geometry Construction



(Close to Romania - OpenStreetMap)

Degrees to Distance

The number of meters one has to travel to move 1 degree in latitude is,

$$111132.92 - 559.82\cos(2\phi_{lat}) + 1.175\cos(4\phi_{lat}) - 0.0023\cos(6\phi_{lat})$$

The same to move 1 degree in longitude is,

$$111412.84\cos(4\phi_{long}) - 93.5\cos(3\phi_{long}) + 0.118\cos(5\phi_{long})$$

[4]

Cubic Splines

The points need to represented as,

$$\boldsymbol{P}_i = (x_i, y_i) = (X_i(u), Y_i(u))$$

where $i = 1, \dots, N$ and N is the number of data points. For a cubic polynomial,

$$X_i(u) = a_{x,i} + b_{x,i}(u - u_{i+1}) + c_{x,i}(u - u_{i+1})^2 + d_{x,i}(u - u_{i+1})^3$$
$$Y_i(u) = a_{y,i} + b_{y,i}(u - u_{i+1}) + c_{y,i}(u - u_{i+1})^2 + d_{y,i}(u - u_{i+1})^3$$

where $u \in [u_i, u_{i+1}]$.

Constraints

We need additional constraints to compute all the coefficients

$$X_i(u)^{(1)}\Big|_{u=u_{i+1}} = D_{x,i} = b_{x,i}$$

$$Y_i(u)^{(1)}\Big|_{u=u_{i+1}} = D_{y,i} = b_{y,i}$$

$$X_i(u)^{(1)}\Big|_{u=u_{i}} = D_{x,i+1} = b_{x,i} + 2c_{x,i}(u_{x,i} - u_{x,i+1}) + 3d_{x,i}(u_{x,i} - u_{x,i+1})$$

$$Y_i(u)^{(1)}\Big|_{u=u_{i}} = D_{y,i+1} = b_{y,i} + 2c_{y,i}(u_{y,i} - u_{y,i+1}) + 3d_{y,i}(u_{y,i} - u_{y,i+1})$$

Practical computation of constraints

- How to choose D_i's?
- Natural splines (special case of Hermite Interpolation) impose

$$X_i^{(2)}(u)\Big|_{u=u_{x,i}} = X_{i+1}^{(2)}(u)\Big|_{u=u_{x,i+1}}$$
$$Y_i^{(2)}(u)\Big|_{u=u_{y,i}} = X_{i+1}^{(2)}(u)\Big|_{u=u_{y,i+1}}$$

- We have $2(N-2)$ equations. We need 4 more conditions to compute D_i's $\forall i = 1, \dots, N$

End conditions for practical computations

- End conditions
- *Clamped or complete*, *not-a-knot*, *periodic*, ...
- Choice of the user and based on application.
- MATLAB's `csapi` from Curve Fitting Toolbox uses *not-a-knot* end conditions.

$$X_2^{(3)}(u)\Big|_{u=u_{x,2}} = X_{N-1}^{(3)}(u)\Big|_{u=u_{x,N-1}} = 0$$

$$Y_2^{(3)}(u)\Big|_{u=u_{y,2}} = Y_{N-1}^{(3)}(u)\Big|_{u=u_{y,N-1}} = 0$$

- $2N$ equations to solve for $D_{x,i}$ and $D_{y,i}$ for $i = 1, \ldots, N$.

(R.H Bartels et al.[3])

Parametrizations

- How to choose the parameter values (knots), u?
- The parameters are recursively computed by,

$$u_0 = 0, \; u_{i+1} = u_i + d_i$$

where we define

$$d_i := \|\boldsymbol{P}_{i+1} - \boldsymbol{P}_i\|_2^{\mu}$$

μ is called the blending parameter. The value of μ influences the overall shape [6].

- $\mu = 0 \implies$ *uniform parametrization* and $\mu = 1 \implies$ *chordal parametrization*.

Normal projection

Coordinates of the boundaries (inner and outer) of the road with width w can be computed by moving in normal direction,

$$\boldsymbol{P}_{i-inner} = \boldsymbol{P}_i - \frac{w}{2}\hat{\boldsymbol{n}}$$

$$\boldsymbol{P}_{i-outer} = \boldsymbol{P}_i + \frac{w}{2}\hat{\boldsymbol{n}}$$

Road modeling

Algorithm

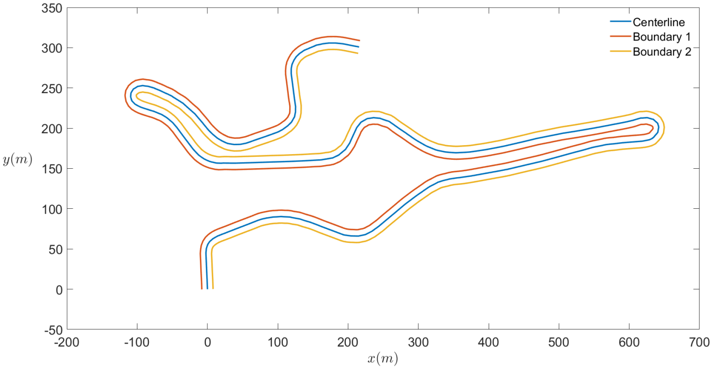
1. Project GPS coordinates of the road centerline.
2. Interpolate the centerline using `csapi`.
3. Sample M points from the centerline spline. Compute the coordinates of inner and outer boundaries of the road.
4. Interpolate the boundaries of the road.

Insights

Findings

1. $\mu = 0 \implies$ poor modeling of the road boundaries when $M >> N$. Inner road boundaries have normal crossings and require separate cleaning and/or smoothing procedure.
2. $\mu = 0$ better in the above aspect.
3. But for a given M that is close to N, $\mu = 0 \implies$ relatively better construction of the overall geometry of the road.

Constructed Road



Modified Road Constraints for the OCP

- Road constraints

$$s(t, z(t)) := \begin{bmatrix} y(t) - y_{up}(x(t)) \\ y_{down}(x(t)) - y(t) \end{bmatrix} \leq 0_{\mathbb{R}^2}$$

- Need to modify our state variable ODEs to include the spline parametrization.

$$\dot{x} = \frac{dx}{du}\dot{u} = t_f v_x \tag{7}$$

$$\dot{y} = \frac{dy}{du}\dot{u} = t_f v_y \tag{8}$$

We can replace the dynamics ODEs (Eqs. 7-8) with the dynamics of parameter t

$$\dot{u} = t_f v_x \left(\frac{dx}{du}\right)^{-1} \tag{9}$$

Modified Road Constraints for the OCP

Modified Road constraints

$$s(t, z(t)) := \begin{bmatrix} y(u) - y_{outer}(u) \\ y_{inner}(u) - y(u) \end{bmatrix} \leq 0_{\mathbb{R}^2} \tag{10}$$

Tests on Straight road

Tests on Straight road

- Feasibility studies were performed on a straight road.
- ODEs were integrated using Explicit Euler method, MATLAB's `ode45` [13] and `dop54` [1]
- Stiff Solvers like MATLAB's `ode15s` and `radau` were also tested - Computationally expensive and were unsuccessful in finding even a feasible point.

Explicit Euler

- Total Time steps N = 40
- $x_0 = 0$ $x_{END} = 140$ m
- interior-point in MATLAB's `fmincon`

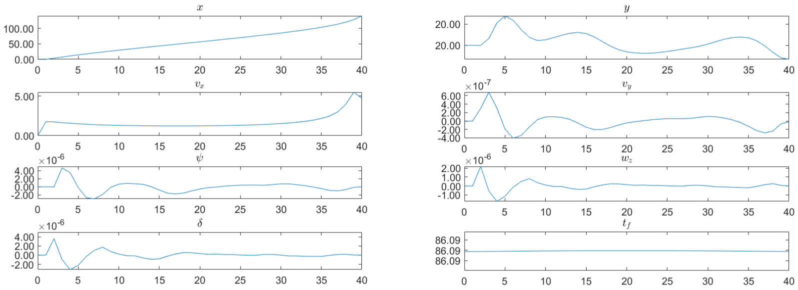
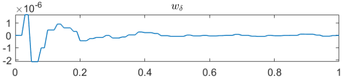


Figure: Feasible State Variables for straight road using Explicit Euler method

Explicit Euler



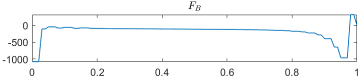


Figure: Feasible Control Variables for straight road using Explicit Euler method

- Feasible solution used as initial guess for optimization.
- Optimal solution not found.

Explicit Runge-Kutta Methods

- Total Time steps N = 30
- $x_0 = 0 \; x_{END} = 2000$ m
- interior-point in MATLAB's `fmincon`

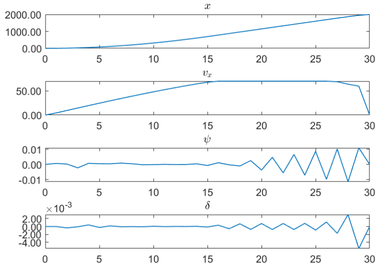


Figure: Feasible State Variables for straight road using `ode45`

Explicit Runge-Kutta Methods

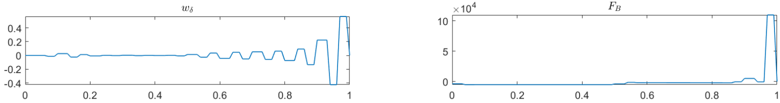


Figure: Feasible Control Variables for straight road using `ode45`

- Feasible solution used as initial guess for optimization.
- Optimal solution not found.

Explicit Runge-Kutta Methods

- Total Time steps N = 30
- $x_0 = 0\ x_{END} = 140$ m
- interior-point in MATLAB's `fmincon`

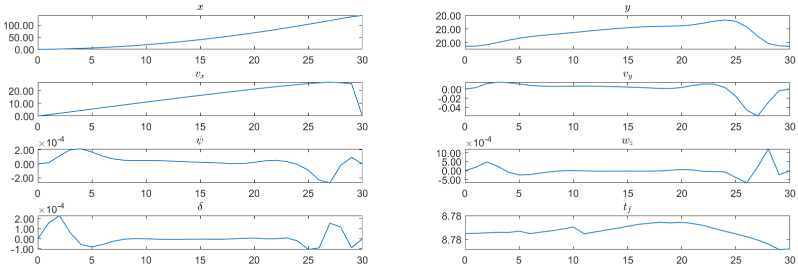


Figure: Feasible State Variables for straight road using `dop54`

Explicit Runge-Kutta Methods

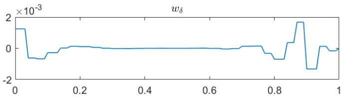


Figure: Feasible Control Variables for straight road using `dop54`

- Feasible solution used as initial guess for optimization.
- Optimal solution not found.

Alternate optimal control problem formulation using Lagrange collocation

Optimal trajectory using Lagrange collocation and new objective formulation

Problems with previous formulation

- Relies on explicit ODE solver
- directly minimising time which is included as a state
- optimization steps are slow because of slow ODE solver computation
- some ODE solvers run into discontinuities during optimization

New objective formulation

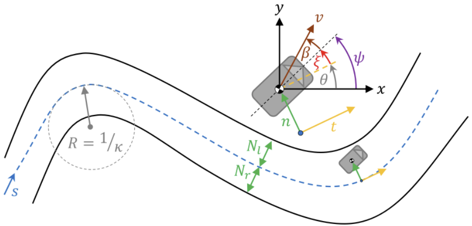
Uses a path coordinate s for formulating ODEs and optimization problem. [12]
s: independent variable
t_f: dependent variable

- s: progress of car along track
- track boundaries, initial and final end points can be specified easily with s
- s not required as state in optimization

New objective formulation

Introduce new variables [5]

- s : parameter for progress along track centerline
- θ : angle between tangent to track centerline and x-direction
- n : lateral displacement from track centerline
- ξ : angle between vehicle axis and tangent to centerline
- κ: curvature of track centerline at any point s



Using s, n, ξ position and orientation of vehicle on the track can be specified

New objective formulation

ODEs for the new states s, n, ξ w.r.t time (t)

$$\dot{s} = \frac{vcos(\xi - \beta)}{1 - n\kappa(s)} \tag{11}$$

$$\dot{n} = vsin(\xi - \beta) \tag{12}$$

$$\dot{\xi} = \omega_z - \kappa \frac{vcos(\xi - \beta)}{1 - n\kappa(s)} \tag{13}$$

New objective formulation

ODEs for the new formulation

$$\dot{v} = \frac{1}{m}\left[(F_{lr} - F_{Ax})cos\beta - (F_{sr} - F_{Ay})sin\beta + F_{lf}cos(\delta+\beta) - F_{sf}sin(\delta+\beta)\right]$$

$$\dot{\beta} = \omega_z - \frac{1}{mv}\left[(F_{lr} - F_{Ax})sin\beta + (F_{sr} - F_{Ay})cos\beta + F_{lf}sin(\delta+\beta) + F_{sf}cos(\delta+\beta)\right]$$

$$\dot{\omega_z} = \frac{1}{I_{zz}}\left[F_{sf}.l_f.cos\delta - F_{sr}.l_r + F_{sf}.l_f.sin\delta\right]$$

$$\dot{s} = \frac{vcos(\xi-\beta)}{1-n\kappa(s)}$$

$$\dot{n} = vsin(\xi-\beta)$$

$$\dot{\xi} = \omega_z - \kappa\frac{vcos(\xi-\beta)}{1-n\kappa(s)}$$

New objective formulation

The ODEs, constraints and objectives are then modified according to new independent variable s. from (1),

$$\frac{dt}{ds} = \frac{1 - n\kappa(s)}{vcos(\xi - \beta)} := J(s) \tag{14}$$

and we get the new objective function by,

$$t_f = \int_{t_0}^{t_f} dt = \int_{s_0}^{s_f} J(s).ds \tag{15}$$

where s_f and s_0 are the values of s at the start and end point of the track.

ODE transformation

Similarly ODEs for all states are transformed for the variable s, for an arbitrary state x and control u,

$$\dot{x} = \frac{dx}{dt} = f(x(t), u(t))$$

is transformed into

$$\dot{x} = \frac{dx}{ds} = \frac{dx}{dt}\frac{dt}{ds} = J(s).f(x(s), u(s)) \tag{16}$$

ODEs of the STM and orientation are transformed using (16)

New optimal control problem formulation

$$
\begin{aligned}
&\min_{x,u} \int_{s_0}^{s_f} J(s).ds \\
&\text{s.t. } \frac{dx}{ds} - f(x(s), u(s)) = 0 \\
&g(x(s), u(s)) \leq 0 \\
&h(x(s), u(s)) = 0 \\
&\gamma(x(s_0), x(s_f)) = 0
\end{aligned}
$$

with states x, controls u, STM equations for v, β, ω_z and orientation equations for n, ξ in f, equality constraints in h, inequality constraints in g and initial and final point constraints in γ

Discretization of states using Lagrange collocation

Controls and states are discretized on a grid $s_k, k = 0, 1, \ldots, N$ with uniform interval Δs where $\Delta s = \frac{s_f - s_0}{N}$

The states on an interval $[s_k, s_{k+1}]$are approximated using Lagrange basis polynomials $p_{k,i}$

$$\bar{x}_k(s) = \sum_{i=0}^{d} \theta_{k,i}.p_{k,i}(s), \;\; s \in [s_k, s_{k+1}]$$

$$p_{k,i}(s) = \prod_{j=0, j\neq i}^{d} \frac{s - s_{k,j}}{s_{k,i} - s_{k,j}}$$

Here $s_{k,i}$ are the collocation points for the interval. We can see that $x_k(s_{k,i}) = \theta_{k,i}, \;\; i = 0, 1, \ldots, d$

Discretization of states using Lagrange collocation

We use Gauss-Legendre collocation with $d = 3$. The controls are discretized using piecewise constant functions. The transformed optimal control problem is

$$\min_{\theta,u} \sum_{k=0}^{N-1} \sum_{i=0}^{d} J_k(\theta_{k,i}) \quad \text{s.t.} \tag{17}$$

$$\dot{\bar{x}}_k(s_{k,j}) - \Delta s f(\theta_{k,j}, u_k) = 0,\ \ k = 0, \dots, N-1, j = 1, \dots, d \tag{18}$$

$$\dot{\bar{x}}_k(s_{k+1,j}) - \theta_{k+1,0} = 0,\ \ k = 0, \dots, N-1 \tag{19}$$

$$g(\theta_k, 0, u(k)) \leq 0, k = 0, \dots, N-1 \tag{20}$$

$$h(x(s), u(s)) = 0,\ \ k = 0, \dots, N-1 \tag{21}$$

$$\gamma(\theta_{0,0}, \theta_{N,0}) = 0 \tag{22}$$

with $\int_{s_0}^{s_f} J(s).ds = \sum_{k=0}^{N-1} \Delta s \sum_{r=0}^{d} B_r J_k(\theta_{k,r})$ where B_r is the integral of the Lagrange basis polynomials.

Optimization for optimal trajectory

We have 6 states $v, \beta, \omega_z, , , \epsilon$ and 3 controls $\delta, F_{drive}, F_{brake}$ with the additional constraint $F_{drive}.F_{brake} = 0$
Road boundary constraints are included using the smooth interpolation described above.
The decision variables of the optimization are
$\{\theta_{0,0}, \ldots, \theta_{0,d}, u_0, \theta_{1,0}, \ldots, \theta_{1,d}, u_1, \ldots, \theta_{N-1,0}, u_{N-1}\}$
where d=3, $\theta_{k,i} \in \mathbb{R}^6$ and $u_k \in \mathbb{R}^3$

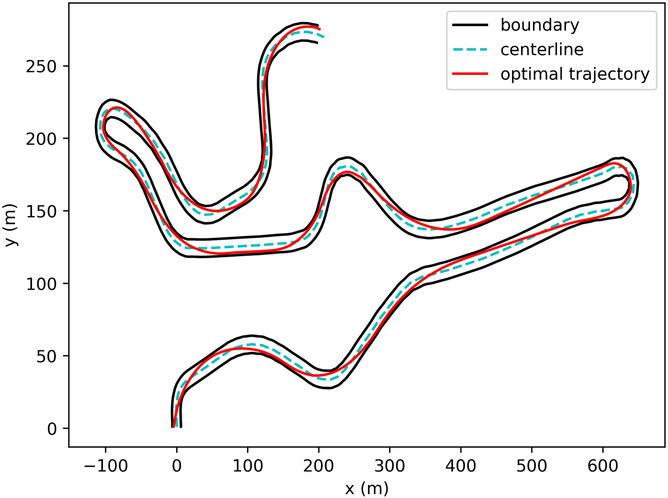
Large number of variables for optimization but analytical solutions of ODEs are explicitly included as equality constraints and ODEs don't require integration by solvers.

We use Primal-Dual Interior Point method to solve the NLP with the solver IPOPT [14] and discretization interval $\Delta s = 3m$.

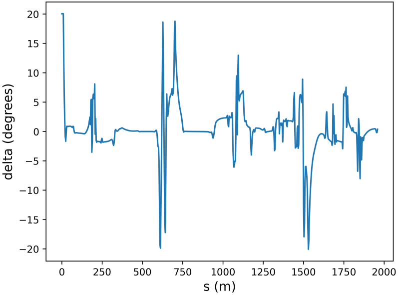
Optimal trajectory

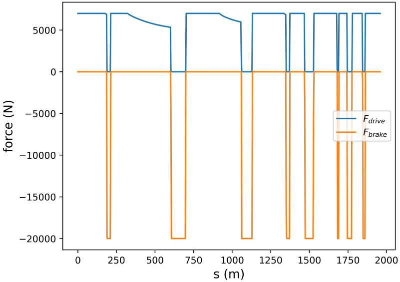
Optimal trajectory time: 78.052 s

Optimization runtime: 19 s

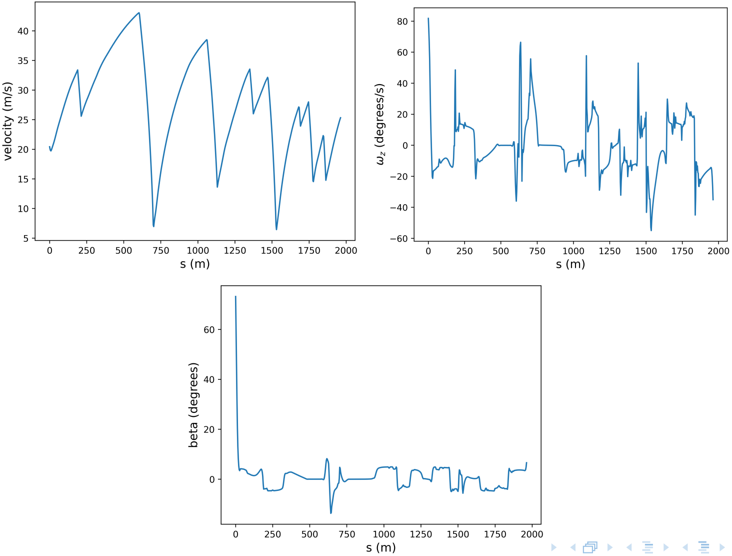


Optimal controls





Optimal states



Model Predictive Control

Model Predictive Control (MPC)

What is a MPC?

- Optimal control strategy

How does MPC work?

- The future response is predicted by predictive control that use a discrete linear time invariant dynamic model

Why is MPC good?

- Control linear and nonlinear systems while taking into account state as well as input constraints

MPC Formulation

$$x_{t+1} = Ax_t + Bu_t$$
$$y_t = Cx_t + u_t$$
$$x_{t+1} \in \mathbb{R}^n, u_{t+1} \in \mathbb{R}^m, y_t \in \mathbb{R}^p, t \in \mathbb{Z}$$

with state vectorx_{t+1},control inputu_{t+1},output vector y_t and time index t

MPC formulation

$$\min_{u} J(x(t), \mathbf{u})$$

$$\mathbf{u} = (u_t, \ldots u_{t+N_p-1})$$

Subject to

$$\begin{aligned} x_{t+k+1} = A_{x_t+k} + B_{u_t+k}, \forall k = 0, \ldots . N_p - 1 \\ x_{t+k} \in \mathbb{X}, \forall k = 0, \ldots . N_p - 1 \\ u_{t+k} \in \mathbb{U}, \forall k = 0, \ldots . N_p - 1 \\ x_t = x(t) \end{aligned}$$

MPC Formulation

Based on the dynamic ODEs

$$\dot{x} = vcos(\psi - \beta)$$
$$\dot{y} = vsin(\psi - \beta)$$

$\ldots$

$$\dot{v} = \frac{1}{m}\left[(F_{lr} - F_{Ax})cos\beta - (F_{sr} - F_{Ay})sin\beta + F_{lf}cos(\delta + \beta) - F_{sf}sin(\delta + \beta)\right]$$

the state space representation is: $\begin{cases} \dot{x} = f(x(t), u(t)) \\ y_t = g(x(t)) \end{cases}$

with state vector $x = [\dot{x}, \dot{y}, \dot{\phi}, \phi, X, Y]$

MPC for Self-Driving Car

Motion Control

- Motion Control MPC is based on vehicle kinetic model by finding the next closest point to plan suitable motion.[15]

Dynamic Control

- Dynamic Control MPC is based on state error composed by heading error, lateral error and location error etc.

$$\text{station error} = -(dx \cos\theta_{des} + dy \sin\theta_{des})$$

$$\text{speed error} = V_{des} - V \cos\Delta\theta/k$$

MPC Formulation

With the objective to follow a desired path, the MPC was formulated as[9]:

$$\min \sum_{t=1}^{N_p} Q_1 e_{y_t}^2 + Q_2 e_{\phi_t}^2 + Q_3 e_{\epsilon_t}^2 + Q_4 u_{t-1}^2 + R_1 \Delta u_t^2$$

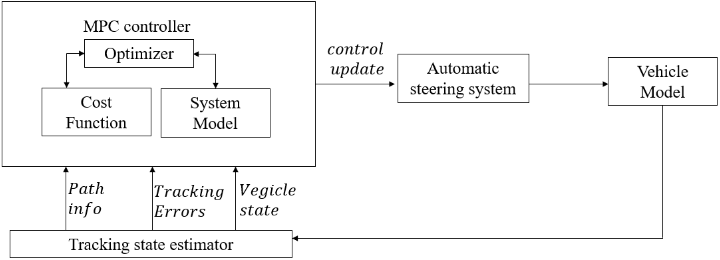
s.t

$$x_{t+1} = Ax_t + B(u_t + \Delta u_t) + C, \forall t = 0, ...N_p$$

$$|u_t| - \varepsilon_t \leq u_{max}, \forall t = 0, ...N_p$$
$$|\Delta u_{t-1}| \leq \Delta u_{max}, \forall t = 0, ...N_p$$
$$\epsilon_t \geq 0, \forall t = 0, ...N_p$$

where e_{y_t} is lateral deviation to the reference path ,e_{ϕ_t} is the error in the heading angle and ϵ_t is a slack variable to soft u_t from becoming infeasible.

MPC Flow Chart



Basic NMPC algorithm for time varying reference x^{ref} [1]

At each simpling time $t_n, n = 0, 1, 2, \ldots$

1. Measure the state $x(n) \in X$ of the system
2. Set $x_0 = x(n)$, solve the optimal control problem

$$\min J_N(n, x_0, u(\cdot)) := \sum_{k=0}^{N-1} \ell(n+k, x_u(k, x_0), u(k))$$

w.r.t $u(\cdot) \in \mathbb{U}^N(x_0)$

subject to $x_u(0, x_0) = x_0, x_u(k+1, x_0) = f(x_u(k, x_0), u(k))$

   and denote the obtained optimal control sequence by $u^*(\cdot) \in \mathbb{U}^N(x_0)$
3. Define the NMPC-feedback value $\mu_n(n, x(n)) := u^*(0) \in U$ and use this control value in the next sampling period.

[1] [8] Lars Grüne and Jürgen Pannek, Nonlinear Model Predictive Control, 2017

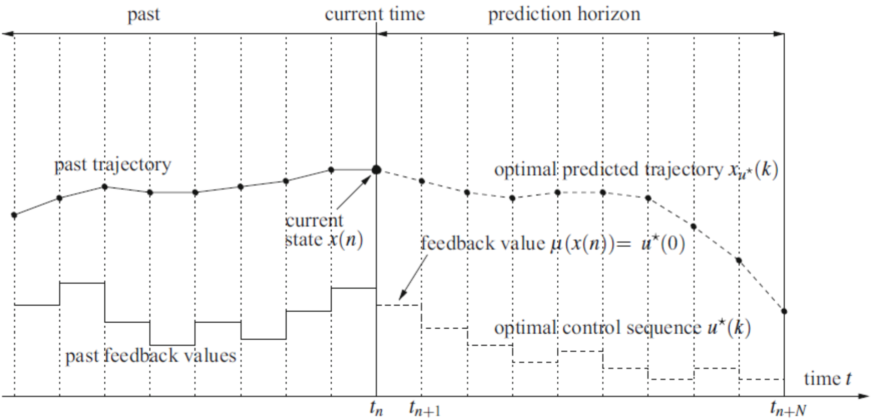


Figure: Illustratin of the NMPC step at time t_n [2]

[2] [8] Lars Grüne and Jürgen Pannek, Nonlinear Model Predictive Control, 2017

From OCP to NLP

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} v\cos\theta \\ v\sin\theta \\ \omega \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} v \\ \omega \end{bmatrix}$$

system model:

$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}_{\boldsymbol{u}}(k), \boldsymbol{u}_k)$ $\quad$ $\boldsymbol{x}(k+1) = \boldsymbol{f}(\boldsymbol{x}(k), \boldsymbol{u}_k)$

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} v\cos\theta \\ v\sin\theta \\ \omega \end{bmatrix} \xrightarrow[\text{Sampling Time } \boldsymbol{\Delta T}]{\text{Euler Discretization}} \begin{bmatrix} x(k+1) \\ y(k+1) \\ \theta(k+1) \end{bmatrix} = \begin{bmatrix} x(k) \\ y(k) \\ \theta(k) \end{bmatrix} + \Delta T \begin{bmatrix} v(k)\cos\theta(k) \\ v(k)\sin\theta(k) \\ \omega(k) \end{bmatrix}$$

running costs: $\ell(\boldsymbol{x}, \boldsymbol{u}) = \|\boldsymbol{x}_{\boldsymbol{u}} - \boldsymbol{x}^{\text{ref}}\|_Q^2 + \lambda\|\boldsymbol{u} - \boldsymbol{u}^{\text{ref}}\|_R^2$

cost function: $J_N(\boldsymbol{x}_0, \boldsymbol{u}) = \sum_{k=0}^{N-1} \ell(\boldsymbol{x}_{\boldsymbol{u}}(k), \boldsymbol{u}(k))$

value function: $V_N(x) = \min_u J_n(\boldsymbol{x}_0, \boldsymbol{u})$

`h =0.2; %[s]` `h =0.4; %[s]`

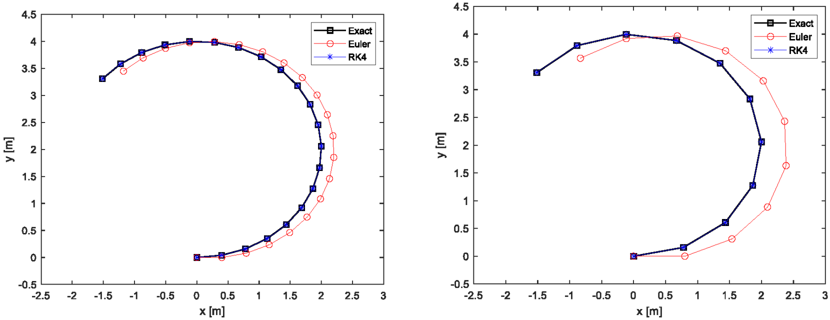
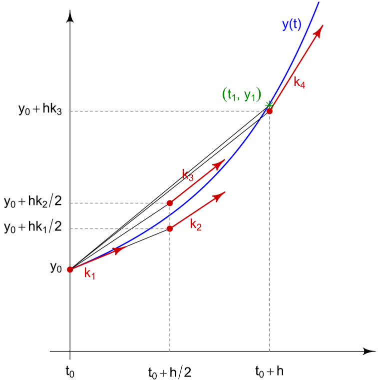


Figure: Comparison between Euler and RK4 [3]

[3] [10] Mohamed W. Mehrez, Prediction Model Simulation Using Runge Kutta Method, 2020

Runge-Kutta 4th

$$\dot{x} = f(x, u(t))$$
$$k_1 = f(x_{n-1}, u)$$
$$k_2 = f(x_{n-1} + \frac{h}{2}k_1, u)$$
$$k_3 = f(x_{n-1} + \frac{h}{2}k_2, u)$$
$$k_4 = f(x_{n-1} + hk_3, u)$$
$$x_n = x_{n-1} + \frac{h}{6}(K_1 + 2k_2 + 2k_3 + k_4)$$



wikipedia: Runge-Kutta Method

From OCP to NLP

$$\textbf{OCP} \xrightarrow[\text{Euler, RK4}]{\text{Shooting Methods}} \textbf{NLP}$$

OCP

$$\begin{aligned}
\min_{u} J_N(\boldsymbol{x}_0, \boldsymbol{u}) &= \sum_{k=0}^{N-1} \ell(\boldsymbol{x}_{\boldsymbol{u}}(k), \boldsymbol{u}(k)) \\
\text{subject to } \boldsymbol{x}_{\boldsymbol{u}}(k+1) &= \boldsymbol{f}(\boldsymbol{x}_{\boldsymbol{u}}(k), \boldsymbol{u}_k), \\
\boldsymbol{X}_{\boldsymbol{u}}(0) &= \bar{\boldsymbol{x}}_0, \\
\boldsymbol{u}(k) &\in U, \forall k \in [0, N-1] \\
\boldsymbol{x}_{\boldsymbol{u}}(k) &\in X, \forall k \in [0, N]
\end{aligned}$$

From OCP to NLP

NLP with $\boldsymbol{w} = [\boldsymbol{u}_0 \cdots \boldsymbol{u}_{N-1}, \boldsymbol{x}_0 \cdots \boldsymbol{x}_N]$

$$\min_{w} \Phi(\boldsymbol{w}) \text{ objective function}$$

$$\text{subject to } \boldsymbol{g}_1(\boldsymbol{w}) = \begin{bmatrix} g_1(\boldsymbol{x}_0, \boldsymbol{u}_0) \\ \vdots \\ g_1(\boldsymbol{x}_{N-1}, \boldsymbol{u}_{N-1}) \\ g_1(\boldsymbol{x}_N) \end{bmatrix} \leq 0, \quad \boldsymbol{g}_2(\boldsymbol{w}) = \begin{bmatrix} \bar{\boldsymbol{x}}_0 - \boldsymbol{x}_0 \\ \boldsymbol{f}(\boldsymbol{x}_0, \boldsymbol{u}_0) - \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{f}(\boldsymbol{x}_{N-1}, \boldsymbol{u}_{N-1}) - \boldsymbol{x}_N \end{bmatrix} = 0$$

inequality constraints — equality constraints

CasADi [4]

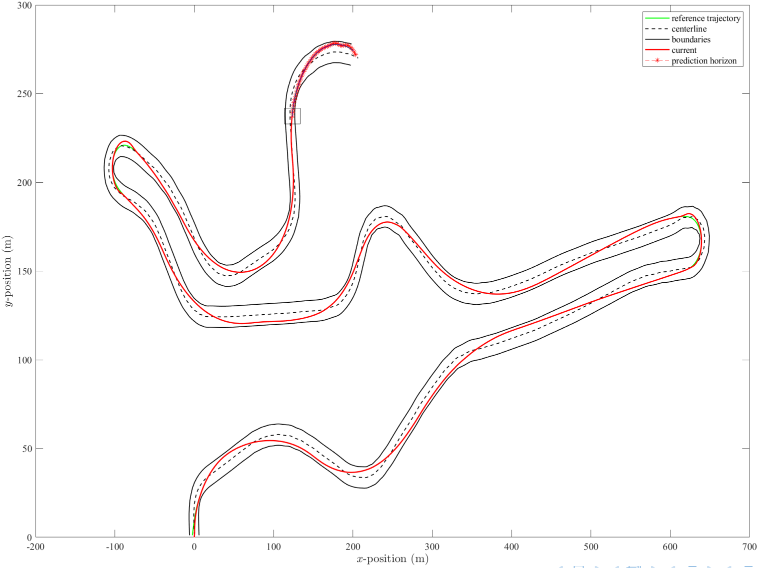
- General scope of numerical optimization
- The solution of NLP's
- Facilitates the solution of optimal control problems (OCPs)
  - Write state-of-the-art OCP algorithms with very little code!
- Free & open-source (LGPL), also for commercial use
- Four standard problems can be handled by CasADi
- Supports high-level operations: matrix-operations, implicit functions, calls to DAE integrators
  - Quadratic programs
  - nonlinear programs
  - root finding problems
  - initial-value problems in ODE/DAE

[4] [2] Joel Andersson, introduction to casadi, 2015

Our Result without obstacles

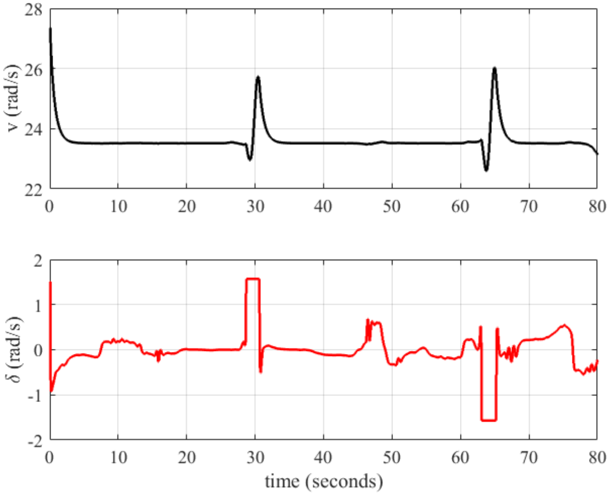
Different choice of prediction horizon (matlab sample).

N=50 in this case.



MPC Simulation

Control inputs for N=50.



Future scope

For Vehicle Dynamics:

- Implement Double Track Model
- Compare the differences

For optimal trajectory:

- Smoothing for control variables
- Friction maps

For MPC:

- Implement more complicated Vehicle Dynamics
- Implement different scenerio
- Proof the stability and robustness of MPC

References I

[1] Roger Alexander. “Solving ordinary differential equations i: Nonstiff problems (e. hairer, sp norsett, and g. wanner)”. In: *Siam Review* 32.3 (1990), p. 485.

[2] Joel Andersson. “Introduction to CasADi”. In: (2015).

[3] Richard H Bartels, John C Beatty, and Brian A Barsky. *An introduction to splines for use in computer graphics and geometric modeling*. Morgan Kaufmann, 1995.

[4] Lev M Bugayevskiy and John Snyder. *Map projections: A reference manual*. CRC Press, 1995.

References II

[5] Fabian Christ et al. “Time-optimal trajectory planning for a race car considering variable tyre-road friction coefficients”. In: *Vehicle System Dynamics* 59.4 (2021), pp. 588–612. DOI: 10.1080/00423114.2019.1704804. eprint: https://doi.org/10.1080/00423114.2019.1704804. URL: https://doi.org/10.1080/00423114.2019.1704804.

[6] Michael S. Floater and Tatiana Surazhsky. “Parameterization for Curve Interpolation”. In: *Studies in Computational Mathematics, Elsevier* 12 (2006), pp. 39–5.

[7] Matthias Gerdts. “Optimal Control of ODEs and DAEs”. In: *De Gruyter* (2012), pp. 366–372.

References III

[8] Lars Grüne and Jürgen Pannek. *Nonlinear model predictive control*. Switzerland: Springer, 2017. DOI: https://doi.org/10.1007/978-3-319-46024-6_3.

[9] Zulkarnain Ali Leman et al. “Model Predictive Controller for Path Tracking and Obstacle Avoidance Manoeuvre on Autonomous Vehicle”. In: *2019 12th Asian Control Conference (ASCC)*. 2019, pp. 1271–1276.

[10] Mohamed W. Mehrez. “Prediction Model Simulation Using Runge Kutta Method”. In: (2020).

References IV

[11] Hans B. Pacejka and Egbert Bakker. “THE MAGIC FORMULA TYRE MODEL”. In: *Vehicle System Dynamics* 21.sup001 (1992), pp. 1–18. DOI: 10.1080/00423119208969994. eprint: https://doi.org/10.1080/00423119208969994. URL: https://doi.org/10.1080/00423119208969994.

References V

[12] Giacomo Perantoni and David J.N. Limebeer. "Optimal control for a Formula One car with variable parameters". In: *Vehicle System Dynamics* 52.5 (2014), pp. 653–678. DOI: 10.1080/00423114.2014.889315. eprint: https://doi.org/10.1080/00423114.2014.889315. URL: https://doi.org/10.1080/00423114.2014.889315.

[13] Lawrence F Shampine and Mark W Reichelt. "The matlab ode suite". In: *SIAM journal on scientific computing* 18.1 (1997), pp. 1–22.

References VI

[14] Andreas Wächter and Lorenz T. Biegler. "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming". English (US). In: *Mathematical Programming* 106.1 (May 2006). Copyright: Copyright 2008 Elsevier B.V., All rights reserved., pp. 25–57. ISSN: 0025-5610. DOI: 10.1007/s10107-004-0559-y.

[15] Yunda Yan et al. "Nonlinear-disturbance-observer-enhanced MPC for motion control systems with multiple disturbances". In: *IET Control Theory Applications* 14 (Jan. 2020). DOI: 10.1049/iet-cta.2018.5821.