Augmentation of Model Predictive Contouring Control using Learning-based Model Predictive Control

Raghavesh Viswanath
Mechanical Engineering
University of Pennsylvania

Jinyuan Guo
Mechanical Engineering
University of Pennsylvania

Sachin Pullil
Mechanical Engineering
University of Pennsylvania

***Abstract*—In this report, we illustrate an augmented Model Predictive Contouring Controller (MPCC) on the basis of Learning-based Model Predictive Control (LMPC). This controller was designed within the Model Predictive Control (MPC) framework, where combined with the non-linear vehicle model and constraints, an optimization problem to provide the vehicle with future steering and throttle values is formulated and solved in an online fashion. The MPCC approach depends on the track center line as the reference trajectory, which does not lead to optimal trajectory generation. We thus apply LMPC to generate an optimal racing line as the reference trajectory and command MPCC to follow it.**

I. Introduction

Autonomous driving is an extremely active research field where over the past few decades, several techniques have been proposed for different driving scenarios. Depending on the control task, the vehicle behavior can be modeled using linear or nonlinear equations of motion. Autonomous car racing is a challenging task for control systems due to the need for maneuvering the vehicle at its handling limits and in nonlinear operating regimes. In addition, the continuously changing environments require advanced path-planning mechanisms executed in real time. These dynamics constrain the sampling time to be in the range of tens of milliseconds, which in turn limits the computational complexity of the algorithms. This situation is even more pronounced when the control algorithm is executed on simple, low-power embedded computing platforms.

In this paper, we investigate optimization-based control strategies for the autonomous car racing task around any given track. We focus on methods that can be implemented to run in real-time on embedded control platforms. The proposed controllers operate the car at its friction limits, beyond the linear region of what is typically used in other autonomous driving systems. This task is usually mastered by expert drivers with countless hours of training, whereas our approach just requires a map of the track along with the car's dynamic model.

The model-based control scheme presented in this paper discusses the application Learning-based Model Predictive Control to generate a reference path offline. At this stage, we integrate the racing track as a local convex set and the vehicle dynamics in a learning-based approach. The model learns over multiple iterations the optimal control effort at each point in time and space. Upon convergence, the LMPC model will have generated a path that minimizes lap time for the car.

We then feed that into the Model Predictive Contouring Control model as the reference trajectory to track as the car progresses along the racing path. Prior research work shows MPCC applied in real 1:43 scale autonomous vehicles that travel as fast as 465 km/hr (288 mph) [1], which showcases its reliability in high speed applications. However, one of the drawbacks of it is that it uses the center line as the reference trajectory and the controller is penalized for straying from this center line. This sometimes leads to inefficient paths followed by the race car. In this work, we attempt to overcome this flaw by directing the online controller to stick to the optimal trajectory as defined by the offline LMPC model.

II. Methods

A. Problem Formulation

In this task, the following state and input vector are considered:

$$\mathbf{x} = \begin{bmatrix} v_x & v_y & \omega_z & e_\psi & s & e_y \end{bmatrix}^{\mathrm{T}}$$
$$\mathbf{u} = \begin{bmatrix} \delta & a \end{bmatrix}^{\mathrm{T}}$$

where ω_z, v_x, v_y are the vehicle's yaw rate, longitudinal and lateral velocities. The position of the vehicle is represented in the curvilinear reference frame [2]. s is the distance traveled along the centerline of the track. The states s_ψ and e_y are the heading angle and lateral distance error between the vehicle and the centerline of the track. Finally, δ and a are the steering and acceleration commands.

B. Learning Model Predictive Control (LMPC)

In learning model predictive control (LMPC), first a local learning predictive controller is designed where the terminal cost and constraints is updated at each time step by using the subset of the stored data. In addition, a system identification exploiting kinematic and data from previous iterations is exploited to identify an affine time variant (ATV) model for control [3].

1) Controller Design: The stored data is exploited to construct an approximation to the cost-to-go over the local convex safe set $\mathcal{CL}_l^j(x)$ around x. Here, $\mathcal{CL}_l^j(x)$ denotes the collected K-nearest neighbors to x from the lth to the jth iterations.

$$\mathcal{CL}_l^j(x) = \{\bar{x} \in \mathbb{R}^n : \exists \lambda \in \mathbb{R}^{K(j-l+1)}, \lambda \geq 0, \mathbb{1}\lambda = 1, \\ D_l^j(x)\lambda = \bar{x}\}$$

where $D_l^j(x)$ represents the matrix collecting K-nearest neighbors to x from the lth to the jth iteration.

The local convex Q-function around x is defined as the convex combination of the cost associated with the stored trajectories.

$$\begin{aligned} Q_l^j(\bar{x}, x) &= \min_{\lambda} \mathbf{J}_l^j(x)\lambda \\ \text{s.t.} &\quad \lambda \geq 0, \mathbb{1}\lambda = 1, D_l^j(x)\lambda = \bar{x} \end{aligned}$$

where the $\mathbf{J}_l^j(x)$ vector collects the cost-to-go associated with the K-nearest neighbors to x from the lth to the jth iteration.

The local convex safe set and the local convex Q-function are used to design the controller. At each time t of the jth iteration, the controller solves the following finite-time optimal control problem:

$$\begin{aligned} J_{t:t+N}^j(x_t^j, z_{tj}) = \min_{\mathbf{U}_t^j, \lambda_\mathbf{t}^\mathbf{j}} & \left[\sum_{k=t}^{t+N-1} h(x_{k|t}^j) + \mathbf{J}_l^{j-1}(z_t^j)\lambda_\mathbf{t}^\mathbf{j} \right] \\ \text{s.t.} \quad & x_{t|t}^j = x_t^j \\ & \lambda_t^j \geq 0,\ \mathbb{1}\lambda_t^j = 1,\ D_l^{j-1}(z_t^j)\lambda_t^j = x_{t+N|t}^j \\ & x_{k+1|t}^j = A_{k|t}^j x_{k|t}^j + B_{k|t}^j u_{k|t}^j + C_{k|t}^j \\ & x_{k|t}^j \in \mathcal{X},\ u_{k|t}^j \in \mathcal{U} \\ & \forall k = t, \ldots, t+N-1 \end{aligned}$$

where $\mathbf{U}_t^j = [u_{t|t}^j, \ldots, u_{t+N-1|t}^j] \in \mathbb{R}^{d \times N}$ and the stage cost is

$$h(x) = \begin{cases} 1, & x \notin \mathcal{X}_F \\ 0, & \text{else} \end{cases}$$

This becomes a finite time optimal control problem. The optimal solution to (8) at time t of the jth iteration $\lambda_t^{j\star}, \mathbf{X}_t^{j\star}, \mathbf{U}_t^{j\star}$ are used to compute vector z_t^j which represents a candidate terminal state for the planned trajectory of the LMPC at time t. Finally, the first element of optimal input vector $u_t^j = u_{t|t}^{j\star}$ is applied to dynamic system and the controller runs again on next state $x_{t+1|t+1}^j = x_{t+1}^j$.

2) ATV Model Fitting: Since the dynamic model of the racing car is nonlinear time variant, a decent approximation of the vehicle dynamics is crucial to the control performance. In [3], kinematic model of the car is first derived and then a regression model is trained to approximate the dynamic model where models the evolution of the vehicle's velocities as a function of the input commands.

Instead of first acquiring non-linear dynamic functions then linearizing them, a linear model around x using a local linear regressor is directlt learned. In particular, for $l = \{v_x, v_y, \omega_z\}$, the following regressor vector is computed:

$$\Gamma^l(x) = \underset{\Gamma}{\operatorname{argmin}} \sum_{(k,j) \in I(x)} K\left(\frac{\|x - x_k^j\|_Q^2}{h}\right) y_k^{j,l}(\Gamma)$$

where K is the Epanechnikov kernel function [4] to compute a local linear model around x for the longitudinal and lateral dynamics.

$$K(u) = \begin{cases} 3(1-u^2)/4, & |u| < 1 \\ 0, & \text{else} \end{cases}$$

And h is a hyperparameter, the set of indices $I_l^j(x)$ collects the indices associated with the P-nearest neighbors to the state x and optimizer $\Gamma \in \mathbb{R}^5$,

$$y_k^{j,v_x}(\Gamma) = \|v_{x_{k+1}}^j - \Gamma[v_{x_k}^j, v_{y_k}^j, \omega_{z_k}^j, a_k^j, 1]^\mathrm{T}\|$$

Following the acquired optimal solution from time $t-1$, at each time t of iteration j, the following ATV model is built

$$x_{k+1|t}^j = A_{k|t}^j x_{k|t}^j + B_{k|t}^j u_{k|t}^j + C_{k|t}^j$$

This model is obtained by linearizing kinematic model and evaluating the approximated vehicle's velocity with optimizer Γ.

$$A_{k|t}^j = \begin{bmatrix} \Gamma_{1:3}^{v_x}(\bar{x}_{k|t}^j) & 0 & 0 & 0 \\ \Gamma_{1:3}^{v_y}(\bar{x}_{k|t}^j) & 0 & 0 & 0 \\ \Gamma_{1:3}^{\omega_z}(\bar{x}_{k|t}^j) & 0 & 0 & 0 \\ (\nabla_x f_{e_\psi}(x)|_{\bar{x}_{k|t}^j})^\mathrm{T} \\ (\nabla_x f_s(x)|_{\bar{x}_{k|t}^j})^\mathrm{T} \\ (\nabla_x f_{e_y}(x)|_{\bar{x}_{k|t}^j})^\mathrm{T} \end{bmatrix}$$

$$B_{k|t}^j = \begin{bmatrix} \Gamma_4^{v_x}(\bar{x}_{k|t}^j) & 0 \\ 0 & \Gamma_4^{v_y}(\bar{x}_{k|t}^j) \\ 0 & \Gamma_4^{\omega_z}(\bar{x}_{k|t}^j) \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$C_{k|t}^j = \begin{bmatrix} \Gamma_5^{v_x}(\bar{x}_{k|t}^j) \\ \Gamma_5^{v_y}(\bar{x}_{k|t}^j) \\ \Gamma_5^{\omega_z}(\bar{x}_{k|t}^j) \\ f_{e_y}(\bar{x}_{k|t}^j) - (\nabla_x f_{e_y}(x)|_{\bar{x}_{k|t}^j})^\mathrm{T}\bar{x}_{k|t}^j \\ f_s(\bar{x}_{k|t}^j) - (\nabla_x f_s(x)|_{\bar{x}_{k|t}^j})^\mathrm{T}\bar{x}_{k|t}^j \\ f_{e_\psi}(\bar{x}_{k|t}^j) - (\nabla_x f_{e_\psi}(x)|_{\bar{x}_{k|t}^j})^\mathrm{T}\bar{x}_{k|t}^j \end{bmatrix}$$

C. Model Predictive Contouring Control (MPCC)

Contouring control is used in industrial applications for machine tools in milling and turning or laser profiling. In these tasks, the problem is solve for inputs to control the tool's movement along the reference path which is given only in spatial coordinates; the associated velocities, angles, etc. that are calculated and imposed by the control algorithm. This is different from tracking controllers as the controller has more freedom to determine the state trajectories to follow the given path, for example to schedule the velocity, in which tracking is defined by the reference trajectory.

The contour following problem can be formulated in a predictive control framework to incorporate constraints. In this, we modify the particular formulation of the MPCC framework [1] to develop a high performance controller for autonomous racing, which tries to maximize progress along the given reference trajectory within the given prediction horizon. In our experiments, we use the path generated from LMPC as the reference trajectory and as a measure of progress. The MPCC approach allows for path planning and tracking to be combined into one nonlinear optimization problem. Before posing the problem, we introduce preliminaries such as parameterization of reference trajectories and error measures.

1) Parameterization of Reference Trajectories: The reference trajectory is parameterized by $\theta \in [0, L]$ with L as the total length. We can obtain any point on the reference trajectory $X^{REF}(\theta)$ and $Y^{REF}(\theta)$ by evaluating a third order polynomial for its argument θ. The angle of the tangent to the path at the reference point with respect to the X-Axis is defined as

$$\Phi(\theta) := \arctan \frac{\partial Y^{ref}(\theta)}{\partial Y^{ref}(\theta)}$$

This parameterization leads to an accurate interpolation within the known points of the reference path.

2) Error Measures: To formulate the MPCC model, we need to define error measures on the car's current X,Y position and heading. We decompose the error into contouring error e^c and lag error e^l. Let

$$\theta_P = P(X,Y) := \underset{\theta}{\operatorname{argmin}}(X - X^{ref}(\theta))^2 + (Y - Y^{ref}(\theta))^2$$

be a projection operator on the reference trajectory. Then the orthogonal distance of the car from the reference path is then given by the contouring error

$$\begin{aligned} e^c(X,Y,\theta_P) := \sin(\Phi(\theta_P))(X - X^{ref}(\theta)) - \\ \cos(\Phi(\theta_P))(Y - Y^{ref}(\theta)) \end{aligned}$$

To make it suitable with online optimization, an approximation θ_A of θ_P is introduced. The lag error is

$$e^l(X,Y,\theta_A) := |\theta_A - \theta_P|$$

In order to be independent of the projection operator θ_P, the contouring and the lag error can be approximated as a function of the position X, Y and the approximate projection θ_A

$$\begin{aligned} e^c &\approx \hat{e}^c(X,Y,\theta_A) := \sin(\Phi(\theta_A))(X - X^{ref}(\theta_A)) - \\ &\cos(\Phi(\theta_A))(Y - Y^{ref}(\theta_A)) \\ e^l &\approx \hat{e}^l(X,Y,\theta_A) := -\cos(\Phi(\theta_A))(X - X^{ref}(\theta_A)) - \\ &\sin(\Phi(\theta_A))(Y - Y^{ref}(\theta_A)) \end{aligned}$$

These essentially represent the tangential and orthogonal component of the error between $X^{REF}(\theta)$ and Y^{REF} and positions X and Y.

3) MPCC Problem: With these error measures formulated, we can now develop the model predictive contouring control problem for autonomous racing. The goal is to find the optimal trade-off between the quality of path-following and progress achieved over a finite horizon of N sampling times.

$$\begin{aligned} \min \sum_{k=1}^{N} &\|\hat{e}^c_k(X_k,Y_k,\theta_{A,k})\|^2_{q_c} + \|\hat{e}^l_k(X_k,Y_k,\theta_{A,k})\|^2_{q_l} - \gamma v_k T_s \\ &+ \|\Delta u_k\|^2_{R_u} + \|\Delta v_k\|^2_{R_v} - -\gamma v_0 T_s \\ \text{s.t.} \quad &\theta_0 = \theta \\ &\theta_{A,k+1} = \theta_{A,k} + v_k / T_s \\ &0 \le \theta_K \le L \\ &0 \le v_K \le \bar{v} \end{aligned}$$

The maximization of final progress measure $\theta_{P,N}$ is replaced by $\sum_0^{N-1} v_k T_s$. To ensure an accurate progress approximation and thus a strong coupling between the cost function and the car model, the weight on the lag error $q_l \in \mathbb{R}+$ is chosen high as suggested in [5]. Furthermore, a cost term on the rate of change of the inputs is added to the objective in order to penalize fast changing controls.

III. Results

First, we applied LMPC on three different tracks shown in figures 1, 2, and 3. Since LMPC is an iterative approach, the number of iterations (laps) it takes to converge to the fastest lap time varies for different tracks. We found for tracks with simpler geometric shapes, like oval-shape track, it takes around 27 iterations to converge, while for those with more complex geometric shapes, like goggle-shape and L-shape track, it takes around 38 iterations to converge. As a result of the iterative nature of this approach, its application in real 1:43 autonomous race cars is limited to offline generation of the optimal trajectory and not real-time execution. An issue that was observed was the generation of a trajectory that occasionally exits the local convex set. For example, in the lower left corner of figure 3, the path is planned outside the racetrack. This is likely due to the fact that for the local convex approximation of the racetrack, programming the constraint employs slack variables which make it a softer constraint. To overcome this issue, we would need to verify the validity of the laps after generation or make the constraint more hard at the cost of convergence time or possible failure.

For real-time execution, we utilize Model Predictive Contouring Control on the race car. Figures 4, 5, and 6 show the tracks that we applied MPCC on and the reference trajectories used. We first ran the controller using the center line as the reference trajectory, as in the traditional MPCC approach. Then, we ran the controller on the optimal trajectory generated by the LMPC controller. First, it is clear that MPCC is able to effectively follow a given reference trajectory closely within a short horizon because it tracks both the center line and the optimal path with minimal error. Second, it appears to have crossed the racetrack boundary in certain areas such as the lower left corner of figure 5. The cause for this is likely the

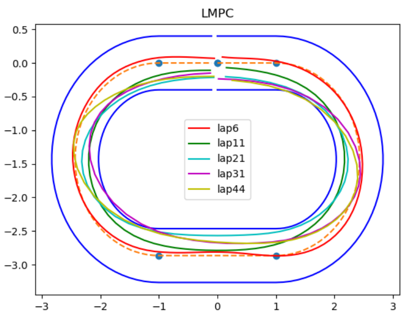

Fig. 1. LMPC on oval-shaped circuit

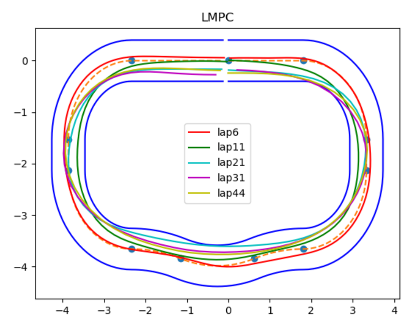

Fig. 2. LMPC on goggle-shaped circuit

same as for LMPC with the application of soft local convex constraints. The adjustments would be the same as discussed earlier. Third, the optimal trajectories generated allow the car to maintain higher accelerations for longer periods of time thus maximizing progress along the track compared to the application of traditional MPCC with the center line as the contour.

IV. Acknowledgement

The authors would like to thank Dr. Michael Posa for his conscientious instruction in the MEAM 5170 course this fall 2022 semester. They would also like to extend thanks to the excellent teaching assistants, Brian Acosta, Alp Aydinoglu, and William Yang for their patience in answering questions throughout the semester.

References

[1] Alexander Liniger, Alexander Domahidi, and Manfred Morari. Optimization-based autonomous racing of 1: 43 scale rc cars. *Optimal Control Applications and Methods*, 36(5):628–647, 2015. 1, 3

[2] Alain Micaelli and Claude Samson. *Trajectory tracking for unicycle-type and two-steering-wheels mobile robots*. PhD thesis, INRIA, 1993. 1

[3] Ugo Rosolia and Francesco Borrelli. Learning how to autonomously race a car: a predictive control approach. *IEEE Transactions on Control Systems Technology*, 28(6):2713–2719, 2019. 1, 2

[4] Vassiliy A Epanechnikov. Non-parametric estimation of a multivariate probability density. *Theory of Probability & Its Applications*, 14(1):153–158, 1969. 2

[5] Denise Lam, Chris Manzie, and Malcolm Good. Model predictive contouring control. In *49th IEEE Conference on Decision and Control (CDC)*, pages 6137–6142. IEEE, 2010. 3

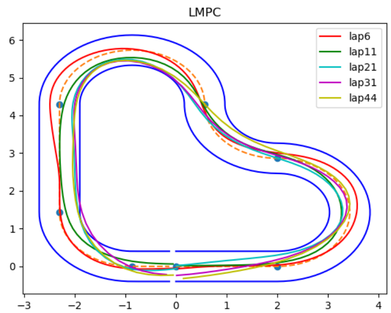

Fig. 3. LMPC on L-shaped circuit

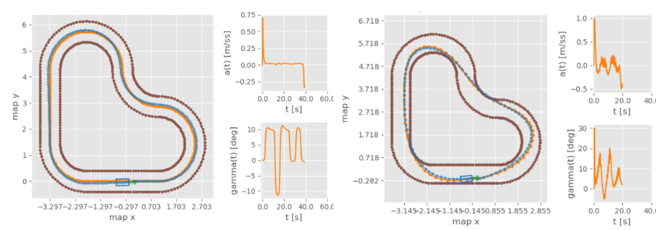

Fig. 4. MPCC on L-shaped circuit on original reference trajectory (left) and LMPC-generated optimal trajectory (right)

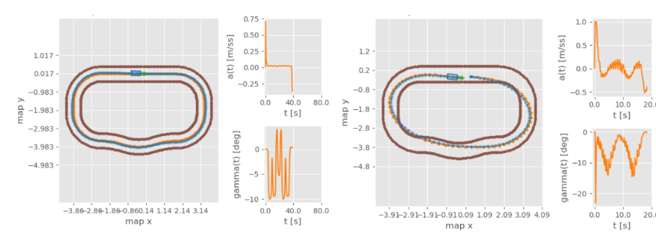

Fig. 5. MPCC on goggle-shaped circuit on original reference trajectory (left) and LMPC-generated optimal trajectory (right)

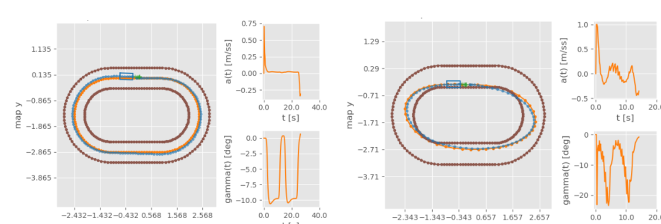

Fig. 6. MPCC on oval-shaped circuit on original reference trajectory (left) and LMPC-generated optimal trajectory (right)