A Modified MPCC Controller with Splines for Autonomous Racing Application

Shikha Tiwari
shikha.tiwari6296@gmail.com

Ankita Pawar
ankitakpawar10@gmail.com

***Abstract*—We present a modified Model Predictive Contouring Control (MPCC) formulation based on splines for miniature autonomous race cars. The method aims at adapting the traditional contouring control formulations for autonomous racing applications. Compared to the standard MPCC controller, the proposed controller is generalized for racing purposes which generates trajectories utilizing the entire track width while making smooth exits out of sharp curvatures on the track subject to non-linear track boundary constraints. Also, the tuning of the controller is simplified for the modified formulation while maintaining the real-time capabilities of the race car. We have validated our proposed controller in simulations and on real race cars.**

I. Introduction

Autonomous car racing is a challenging task for automatic control systems due to the need for handling the vehicle close to its stability limits and in highly nonlinear operating regimes. Fast dynamics constrain the sampling time to be in the range of a few tens of milliseconds at most, which severely limits the admissible computational complexity of the algorithms. This situation is even more challenging if the autonomous algorithms shall be executed on simple, low-power embedded computing platforms [1].

In this report, we build upon an NMPC formulation that combines path planning and path following into one formulation, known as model predictive contouring control (MPCC). MPCC was originally developed for machining application [1] which makes it more suitable for path following, and this original formulation was taken unaltered for autonomous racing application [1], [3], [4]. Traditional MPCC is based on the assumption that there is a small deviation between the car's position and its relative position on the reference path which may not hold for autonomous racing where race tracks feature sharp curves. The controller with this formulation generates trajectories mostly following the reference path (or center line) and also the velocities decrease abruptly when exiting sharp turns.

Our approach toward the problem was to modify the original MPCC formulation such that the controller is targeted towards maximizing the progress of the car on the track while maintaining the car within the track boundaries. Also, we supported this implementation by representing our reference trajectory using splines instead of using a set of sampled points. With this reformulation, the car generates smoother trajectories that exploit larger regions of the track width while being within the track boundaries and also have less number of weights to be tuned compared to the original MPCC. The car's velocities for the majority of the track length are constantly high and therefore result in faster lap time finishes. We support these claims with experiments in simulations and on real race cars.

The report is structured as follows. Section II presents the car model used for the control problem. Section III presents the original MPCC formulation and its limitations and later our modified formulation. Section IV presents the experimental results comparing both formulations and we end the report with Sections V and VI for conclusions and future scope.

II. Car Model

For the MPCC formulation, the race car is modeled using bicycle model [5]- [6] in addition to modeling the tire forces with simplified Pacejka Tire model [7]. The model is given by the following equations,

$$\dot{X} = V_X\ cos(\psi) - V_Y\ sin(\psi), \tag{1a}$$

$$\dot{Y} = V_X\ sin(\psi) + V_Y\ cos(\psi), \tag{1b}$$

$$\dot{\psi} = \omega, \tag{1c}$$

$$\dot{V_x} = \frac{1}{m}(F_{r,X} - F_{f,Y}\ sin\delta + mV_Y\omega), \tag{1d}$$

$$\dot{V_Y} = \frac{1}{m}(F_{r,Y} + F_{f,Y}\ cos\delta - mV_X\omega), \tag{1e}$$

$$\dot{\omega} = \frac{1}{I}(F_{f,Y}\ l_f\ cos\delta - F_{r,Y}\ l_r), \tag{1f}$$

where the state consists of the X, Y positions of the car's CoG (Center of Gravity) in the inertial frame, ϕ is the angle of the car with the inertial frame, V_X and V_Y are the longitudinal and lateral components of car's linear velocity and ω is the yaw rate and θ is arc length of the track covered by the car. The control inputs include the rate of change in throttle ΔT, steering angle $\Delta\delta$, and the arc length $\Delta\theta$ (which as will be explained later is added due to consideration of error approximations). In equations (1a) - (1f) the subscripts X and Y indicate longitudinal and lateral forces or velocities, while r and f refer to the front and rear tires, respectively. Also, lf and lr are the distances for the CoG to the front and the rear wheels respectively [1].

III. Autonomous Racing Controllers

A. Overview of Traditional MPCC Formulation

MPCC is a control scheme based on the minimization of the cost function which reflects the tradeoff between com-

peting objectives of accurate tracking of the geometric path while maximizing the distance traversed within the predicted horizon. MPCC works for high-speed control applications, requiring a convex problem formulation to reduce computational cost [8].

MPCC follows the center line of the track as the reference path which is merely used as a measure of progress on the track by employing lower weights on tracking (contouring) errors. It has advantages since it combines both path planning and path tracking together in a single objective which can be solved in real-time by approximating the NLP using local convex QP approximations at each sampling time [9]. The following sub-sections will elaborate on both its objective and later the final formulation is mentioned.

1) Contouring Control Objective: The objective is used to define the error measure as the difference between the car's current position X, Y from the desired position $X^{ref}(\theta)$ and $Y^{ref}(\theta)$ on the reference path, where θ denotes the arc length of reference path at the closest reference point as shown in Fig.1 [1]. It is expressed as,

$$\begin{aligned} e^c(X,Y,\theta_P) \triangleq\ & sin(\phi(\theta_P))(X - X^{ref}(\theta_P)) \\ & - cos(\phi(\theta_P))(Y - Y^{ref}(\theta_P)) \end{aligned} \tag{2}$$

Contouring error $e^c(X,Y,\theta_P)$ is the orthogonal distance of the car's current position to the reference path, minimizing this will make sure that the generated control signals by the solver will make the car follow the reference curve. The equation for contouring error is further approximated since the estimation of θ_P is in itself another optimization problem. Therefore, θ_P is approximated by θ_A variable which is determined by the controller itself. Also, in practical scenarios where equally distant sampled points are used to represent the reference path, this results in contouring error approximation expressed as,

$$\begin{aligned} \hat{e}^c(X,Y) \triangleq\ & sin(\phi^{ref})(X - X^{ref} - \nabla X^{ref}(\theta_A - \hat{\theta}_A)) \\ & - cos(\phi^{ref})(Y - Y^{ref} - \nabla Y^{ref}(\theta_A - \hat{\theta}_A)) \end{aligned} \tag{3}$$

2) Progress Control Objective: The introduction of θ_A results in another cost term i.e., the lag error. The basic idea here is that there is a point θ_A ahead of the car's current relative position on the reference path from which it lags. The approximation of lag error with reference path representation as sampled points is given as,

$$\begin{aligned} \hat{e}^l(X,Y) \triangleq\ & - cos(\phi^{ref})(X - X^{ref} - \nabla X^{ref}(\theta_A - \hat{\theta}_A)) \\ & - sin(\phi^{ref})(Y - Y^{ref} - \nabla Y^{ref}(\theta_A - \hat{\theta}_A)) \end{aligned} \tag{4}$$

We need to give higher weights to the lag error to ensure its value is small and the approximation of θ_P by θ_A is accurate enough. θ is a linearization point and $\hat{\theta}_A$ is predicted output of last iteration of NLP formulated in (5a). The approximated contouring and lag error is depicted in Fig. 1.

3) Final MPCC Problem: To form the lag error at each time step in the prediction horizon, it is necessary to introduce an integrator state with dynamics $\theta_{A,k+1} = \theta_{A,k} + \Delta\theta_{A,k}$,

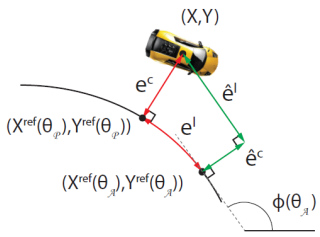

Fig. 1. Contouring error e^c and lag error e^l with linear approximations $\hat{e}_c$ and $\hat{e}_l$ [1]

where $\theta_{A,k}$ is the state of progress at time step k and the variable $\Delta\theta_A$ can be seen as the projected velocity [1]. This approximation results in an optimal control problem in the NLP form as,

$$\begin{aligned} \min_{x,u} \sum_{k=1}^{N} & \|\hat{e}_k^c(X_k,Y_k)\|_{Q_1}^2 + \|\hat{e}_k^l(X_k,Y_k)\|_{Q_2}^2 - q\theta_{A,k} + \\ & \|\Delta T_k\|_{R1}^2 + \|\Delta\delta_k\|_{R2}^2 + \|\Delta\theta_{A,k}\|_{R3}^2 \end{aligned} \tag{5a}$$

subject to

$$x(0) = x_0 \tag{5b}$$

$$x_{k+1} = f(x_k, u_k),\ k = 0,\dots,N-1 \tag{5c}$$

$$\theta_{A,k+1} = \theta_{A,k} + \Delta\theta_{A,k},\ k = 0,\dots,N-1 \tag{5d}$$

$$x_{min} \le x_k \le x_{max},\ k = 1,\dots,N \tag{5e}$$

$$u_{min} \le u_k \le u_{max},,\ k = 1,\dots,N \tag{5f}$$

$$(X_k - X_k^{ref})^2 + (Y_k - Y_k^{ref})^2 \le r^2,\ k = 1,\dots,N \tag{5g}$$

where, approximated contouring error $\hat{e}_k^c(X_k,Y_k)$ is defined as per (3) and lag error $\hat{e}_k^l(X_k,Y_k)$ as per (4), $\Delta T_k \triangleq T_k - T_{k-1}$, and $\Delta\delta_k \triangleq \delta_k - \delta_{k-1}$. The consideration of $\theta_{A,k}$ dynamics results in a penalty term on maximation of the final progress $-q\theta_{A,N}$ which compliments large values of θ_A with reasonable bounds on $\theta_{A,k}$ and $\Delta\theta_{A,k}$ values. Furthermore, a cost term on the rate of change of the control inputs is added to the objective to penalize fast-changing controls, which helps to obtain smooth control inputs to prevent amplifying unmodeled dynamics [1]. Finally, the cost term on projected velocity $\Delta\theta_{A,k}$ is to avoid large linearization errors in the estimation of $\hat{e}^c$ and $\hat{e}^l$ values.

As for the constraints it includes the initial state equality constraint (5b), the vehicle state dynamics constraint (5c) as pacejka model equations (i.e., bicycle model with pacejka tire forces), bounds on the state (5e) and the control variables (5f) are considered as a limit to the physical admissible values, and finally the circular track constraints (5g) approximated as the squared distance between the car's current position (X_k, Y_k) to its relative position on the reference path (X_k^{ref}, Y_k^{ref}).

4) Limitations of Traditional MPCC: As already discussed in the introduction, the original MPCC formulation is based on the assumption that the lateral error between the car's position

and the reference trajectory is very small. This assumption does not hold for racing applications where the optimal racing line is the one in which the entire track width is used to achieve minimal lap times.

We could argue that minimal lap time can be achieved by tuning the weights of the MPCC objective appropriately. But this leads to different issues due to the approximation considered for the θ_A dynamics given in (5c) which is a reasonable approximation in cases when the lateral error is small. If contouring cost is given lower weights, this would mean the lateral error is allowed to be larger and due to inaccurate θ_A dynamics consideration, the resulting θ_A values are inaccurate as well leading to incorrect evaluation of constraints dependent on variable θ_A. While for mid or larger values of contouring cost weights, the vehicle mostly follows the reference path, and also due to high contouring weights the controller tends to decrease the velocity abruptly at curved sections and thereafter introduce non-smoothness in the car trajectory. Also, in this MPCC formulation tuning of the controller is extremely challenging and scenario dependent [10]. Our modified formulation tries to minimize this challenge to some extent.

B. Modified MPCC Formulation with Splines

1) Reference path as splines: Mostly the implementation of MPCC involves optimization subject to closely distanced reference points to assure that the car does not deviate from the path, which is mostly required for trajectory tracking scenarios. This seems to be a contradicting problem formulation and therefore having a method to generate reference points is important [8].

Since the MPCC includes both the tracking accuracy and the maximal progress in its objective we could make use of splines for reference path parameterized by the arc length $\theta \in [0, L]$, where L is the total length of the reference path. The whole reference path is broken into multiple segments where in each segment a polynomial is interpolated between the control points (the ends of the segments) to form a continuous curve. Normally cubic splines are used which are the simplest lowest order piecewise polynomials [11].

The offline fitting of splines on top of the sampled control points is performed to obtain a smooth and continuous representation of the reference path. For this, we have used Python's scipy library for spline fitting and stored the spline coefficients for later use. Referring to Fig. 2, we can observe the sampled control points and the cubic spline interpolation curve parameterized over the arc length θ. By evaluating a third-order polynomial for its argument θ we can obtain any reference point $X^{\text{ref}}(\theta_A)$ and $Y^{\text{ref}}(\theta_A)$ on the reference path of the track and the corresponding tangent angle to the path at the reference point concerning the X-axis given by,

$$X^{ref}(\theta_A) = a_x + b_x\theta_A + c_x\theta_A^2 + d_x\theta_A^3 \tag{6a}$$

$$Y^{ref}(\theta_A) = a_y + b_y\theta_A + c_y\theta_A^2 + d_y\theta_A^3 \tag{6b}$$

$$\phi^{ref}(\theta_A) = \frac{\partial Y^{ref}(\theta_A)}{\partial X^{ref}(\theta_A)} \tag{6c}$$

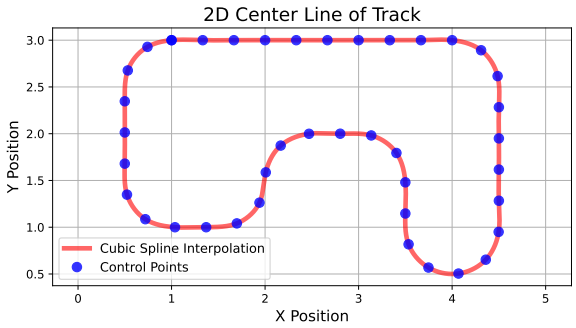

Fig. 2. Result of the cubic spline interpolator fitting across the sampled control points

The splines being continuous and a better approximation of the reference path provide smoother derivatives which aid in the consistency of error and error rate calculations. By using splines, the approximated contouring and lag error is expressed as [1],

$$\begin{aligned}\hat{e}^c(X, Y, \theta_A) \triangleq\ & sin(\phi(\theta_A))(X - X^{ref}(\theta_A)) \\ & - cos(\phi(\theta_A))(Y - Y^{ref}(\theta_A))\end{aligned} \tag{7a}$$

$$\begin{aligned}\hat{e}^l(X, Y, \theta_A) \triangleq\ & -cos(\phi(\theta_A))(X - X^{ref}(\theta_A)) \\ & - sin(\phi(\theta_A))(Y - Y^{ref}(\theta_A))\end{aligned} \tag{7b}$$

2) Modified MPCC formulation: The inclusion of reference path as splines leads to the reduction in the lag errors compared to when equally distant reference points were used. This also makes the approximate θ_A to be rather equal to the projected θ_P and hence the approximated contouring errors and the circular track constraints are almost equivalent. Furthermore, to ensure that the controller's objective is targeted towards maximal progress on the track, the removal of contouring cost from the objective function aids in achieving our purpose. With these considerations, the modified MPCC formulation NLP is expressed below,

$$\begin{aligned}\min_{x,u} \sum_{k=1}^{N-1} & \|\hat{e}_k^l(X_k, Y_k, \theta_{A,k})\|_{Q_2}^2 - q\theta_{A,k} + \|\Delta T_k\|_{R1}^2 + \\ & \|\Delta\delta_k\|_{R2}^2 - q_N\theta_{A,N}\end{aligned} \tag{8a}$$

subject to

$$x(0) = x_0 \tag{8b}$$

$$x_{k+1} = f(x_k, u_k),\ k = 0, \ldots, N-1 \tag{8c}$$

$$\theta_{A,k+1} = \theta_{A,k} + \Delta\theta_k,\ k = 0, \ldots, N-1 \tag{8d}$$

$$x_{min} \leq x_k \leq x_{max},\ k = 1, \ldots, N \tag{8e}$$

$$u_{min} \leq u_k \leq u_{max},,\ k = 1, \ldots, N \tag{8f}$$

$$(X_k - X_k^{ref}(\theta_{A,k}))^2 + (Y_k - Y_k^{ref}(\theta_{A,k}))^2 \leq r^2, \tag{8g}$$

$$k = 1, \ldots, N$$

Here the reformulation considers the lag error cost de-

fined in (7b), in its objective to maintain the accuracy of approximations made using θ_A as was the case in the original formulation, and also it maintains the stability of the controller to avoid spurious NLP solutions. The penalties on the rate of change of control inputs to avoid fast-changing input controls remain the same. Also, the penalty terms on the arc length for getting maximal progress along the reference path are separated into two terms one representing the penalty for arc length over k = 0 to $k = N-1$ horizon intervals i.e. $q\theta_{A,k}$ and a penalty for terminal horizon arc length i.e. $q_N\theta_{A,N}$. This was done since the maximation of the final progress is dependent on the terminal horizon arc length and also this allows us to have separate weighing opportunities for both penalty terms.

As for the constraints most of them remain the same as were for the original formulation such as the initial state condition, the vehicle state dynamics, and the physical limitations or bounds on the states and controls values. Also, the circular track constraint is rather used instead of a complex contouring error equation to make sure that the vehicle avoids exiting the track geometry.

This nonlinear optimization problem is formulated using CasADi [12] and solved in real-time by approximating the NLP using local convex QP approximations at each sampling time using SQP-RTI (Sequential quadratic Program – Real-Time Iterations) as the NLP solver [13] and Gauss-Newton was used for hessian approximations in Acados software framework [14], [15].

Now having removed the contouring cost and the cost on $\Delta\theta_{A,k}$ from the objective, the main focus of the controller is directed towards obtaining maximal progress along the track which is an expected behavior for racing applications. Also, this makes the tuning of the controller simpler due to less number of weights in the modified formulation.

IV. Results

In this section, we compare the performances in simulations and on real cars of the original MPCC and the modified MPCC with splines. For our experiments, we used set up provided by our race car control lab and source code provided by ETH Zurich University. As was observed in the simulation run, the lag errors for the modified controller were moderately lower in the order of 10^{-3} to 10^{-8} than the original MPCC for which errors were in the order of 10^{-1} to 10^{-3}. This was due to the use of cubic splines (i.e., 40 piecewise cubic polynomials for the whole track length) which gave a smoother representation of the reference path as shown in Fig. 2 and comparatively accurate estimation of the relative position of the car on the reference path than using fixed distant sampled reference points representation.

We proceed by comparing the computational performance of both the formulations, as given in Table I, where we tried several combinations of solver frequency (Hz) and horizon length (N) for each solve it can be observed that the modified MPCC takes 4-5 times the computation time of the original MPCC, that is in the range of 20 milliseconds. The reasoning is that the solver is unable to perform structure detection which is a simplification of the expressions for solver functions and variables that hold the piecewise cubic polynomials [15]. The modified MPCC with splines does feature higher computation times, but it can achieve real-time feasibility. Therefore, to get better computation times the solver expressions for the piecewise cubic polynomials need to be improved to get fast solve times. Also, the reduction in the number of cubic polynomials that represent the reference path will reduce the solver computation time. For our experiment, we used 40 cubic polynomials for the reference path, while 20 could have been sufficient.

TABLE I
Solver performance for MPCC formulations

Solver Frequency (Hz)	Horizon Length (N)	Time taken by solver per solve (mSec)	
		Original MPCC Formulation	Modified MPCC Formulation with Spline
25	30	3.181	QP Fails to generate solution
	40	3.498	
	45	3.626	
30	30	3.501	17.756
	40	3.726	19.798
	45	3.987	22.224
40	30	3.622	19.495
	40	3.891	21.752
	45	4.032	24.054

TABLE II
Lap times for MPCC Formulations

MPCC Formulation	Simulation Lap Time (sec)	Real Car Lap Time (sec)
Original MPCC	4.77-4.85	6.14-6.35
Modified MPCC with spline	4.62-5.01	5.24-5.57

Also, referring to the same table for the modified MPCC the QP fails to generate a solution for lower solver frequencies of 25 Hz and less which is majorly due to the combination of state variables bounds and the non-linear constraints that result in infeasibility of QP. Also, at higher solver frequencies that are 40 Hz and above the predicted trajectory shows oscillatory behavior at the straight sections of the track due to low sampling times and reduced lookahead distances.

The best optimal solutions are obtained at a solver frequency of 30 Hz and a horizon length of 40 for both formulations, which corresponds to a 1.3 sec ahead prediction. The weights used for this experiment for change in throttle and change in steering angle i.e., $R_1 = 0.2$ and $R_2 = 0.3$ fixed for both formulations, while for the original MPCC which had contour cost weight as $Q_1 = 100$ and relatively higher weight on the lag error cost i.e., $Q_2 = 200$ with additional weight on change in arc length $\Delta\theta$ i.e., $R_3 = 0.2$, to make sure that car progresses in the forward direction. As can be seen in Fig. 3, the car's predicted trajectory for traditional or original MPCC mostly follows the center line and also is

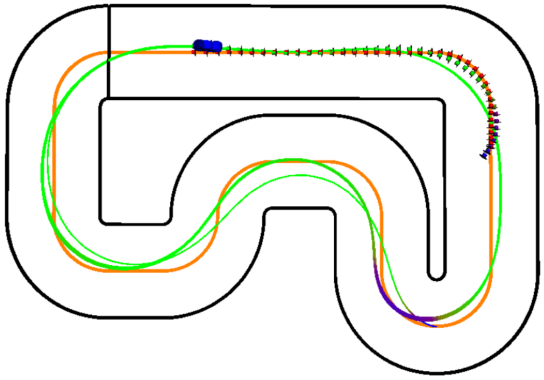

Fig. 3. Simulation optimal trajectory for original MPCC

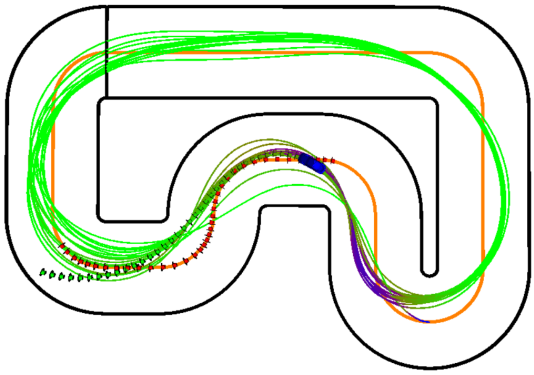

Fig. 4. Simulation optimal trajectory for modified MPCC with splines

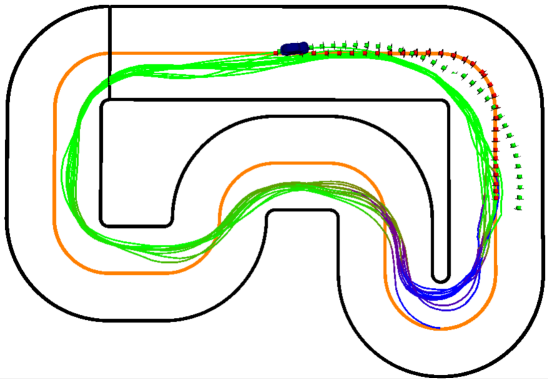

Fig. 5. Real race cars experiment optimal trajectory for modified MPCC with splines

not very smooth when exiting the sharp curved section and also velocities decrease (i.e., 1.5 m/s and below) at the last few predicted horizon points represented in color blue which as we have already discussed in the limitations is because the controller trying to reduce the large lateral errors at the turns. When compared with the modified MPCC which had lag error cost weight as $Q_2 = 1.0$ and $q = 1.0$ and $q_N = 10.0$ weights for the penalties on the arc lengths over the horizon of the solver, which we obtained by trial-and-error experiments with verification of consistency in the car's behavior, it was observed as in Fig. 4 that the car's predicted trajectory for modified MPCC uses larger track width which is much smoother when exiting the curvatures and the velocities remains high for a large section of the track length that is 2.5 m/s and above represented in color green. As for the lap times, Table II depicts that in general the simulation lap times of both the formulations are comparable, except for the fastest lap times values suggest that the modified MPCC was able to produce the better fastest lap times than the original MPCC.

The experiments on the real-world miniature cars with modified MPCC carried at solver frequency of 30 Hz and horizon length (N) of 40 and at the same weights as used in simulation experiments the result is shown in the Fig. 5. As for the solver performance, the time for a single solve was the same as in the simulation run i.e. about 20 - 25 milliseconds. The trajectories may not be as expected which is due to model parameters and track dimension disparities in the simulations and real cars which was also observed for the original MPCC. But on the other side the measured lap times were approximately 14% less for the modified MPCC as mentioned in Table II.

V. Conclusion

This work presented a modified formulation of MPCC with the splines-based representation of the reference path and can produce racing line kind of trajectories. We compared our method with the original MPCC formulation implemented with equally distant sampled points for reference path and proved in simulation and experiments on real cars that our proposed method is simpler for tuning because of the lesser number of weights and, also produces better fastest lap times with reliability.

VI. Future Scope

Experiments with different initial guesses and some globalization techniques such as line search algorithms can be tried to observe the effects. Also, the model parameters used for the experiments can be replaced with optimized parameters to get better performance. More effort needs to be directed toward reducing the computation time of the solver. Also most importantly work towards equating better dynamics for arc length θ is required.

VII. Acknowledgement

We would like to thank Tobias Bodewig for supporting us through the project with his valuable suggestions regarding the limitations of the Original MPCC formulation.

References

[1] Liniger, A., Domahidi, A., and Morari, M. , "Optimization-based autonomous racing of 1: 43 scale RC cars. " Optimal Control Applications and Methods, vol. 36(5), pp. 628–647, July 2015.

[2] D. Lam, C. Manzie, and M. Good, "Model predictive contouring control," in Proceedings of the IEEE Conference on Decision and Control. Institute of Electrical and Electronics Engineers Inc., 2010, pp 6137–6142.

[3] W. Schwarting, J. Alonso-Mora, L. Pauli, S. Karaman, and D. Rus, "Parallel autonomy in automated vehicles: Safe motion generation with minimal intervention," in Proceedings - IEEE International Conference on Robotics and Automation. Institute of Electrical and Electronics Engineers Inc., 7 2017, pp. 1928–1935.

[4] O. De Groot, B. Brito, L. Ferranti, D. Gavrila, and J. Alonso-Mora, "Scenario-Based Trajectory Optimization in Uncertain DynamicEnvironments," IEEE Robotics and Automation Letters, vol. 6, no. 3, pp. 5389–5396, 7 2021.

[5] E. Velenis, E. Frazzoli, and P. Tsiotras, "On steady-state cornering equilibria for wheeled vehicles with drift," in Proceedings of the 48th IEEE Conference on Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference, pp. 3545–3550, IEEE, 2009.

[6] C. Voser, R. Y. Hindiyeh, and J. C. Gerdes, "Analysis and control of high sideslip manoeuvres," Vehicle System Dynamics, vol. 48, no. S1, pp. 317–336, 2010.

[7] E. Bakker, L. Nyborg, and H. Pacejka, "Tyre modelling for use in vehicle dynamics studies," SAE, 1987.

[8] Johan Kellerth Fredlund, Kenan Sadik Sulejmanovic, "Autonomous driving using Model Predictive Control methods," LUUD University, Department of Automatic Control, April 2017.

[9] M. Diehl, H. Bock, J. Schl¨oder, R. Findeisen, Z. Nagy, and F. Allg¨ower, "Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations," Journal of Process Control, vol. 12, no. 4, pp. 577–585, 2002.

[10] Lyons, Lorenzo and Ferranti, Laura , "Curvature-Aware Model Predictive Contouring Control ", 2023.

[11] Kelly, Matthew O', and Hongrui Zheng. "Raceline Optimization." F1TENTH Foundation, Accessed 2 July 2023, f1tenth.org/learn.html.

[12] Joel A E Andersson and Joris Gillis and Greg Horn and James B Rawlings and Moritz Diehl, "CasADi - A software framework for nonlinear optimization and optimal control", Mathematical Programming Computation, vol. 11, no. 1, pp. 1–36, 2019, doi:10.1007/s12532-018-0139-4.

[13] M. Diehl, H. G. Bock, and J. P. Schl¨oder, "A real-time iteration scheme for nonlinear optimization in optimal feedback control," SIAM Journal on Control and Optimization, vol. 43, no. 5, pp. 1714–1736, 2005.

[14] B. Houska and H.J. Ferreau and M. Diehl, "An Auto-Generated Real-Time Iteration Algorithm for Nonlinear MPC in the Microsecond Range", Automatica, 47, 2279–2285, 10, 2011, doi:10.1016/j.automatica.2011.08.020

[15] B. Houska and H.J. Ferreau and M. Vukov and R. Quirynen, ACADO Toolkit User's Manual, 2009–2013.

[16] J. Frey et al., "Detecting and Exploiting Generalized Nonlinear Static Feedback Structures in DAE Systems for MPC," 2019 18th European Control Conference (ECC), Naples, Italy, 2019, pp. 2756-2762, doi: 10.23919/ECC.2019.8795732.

[17] Andrea Carron and Sabrina Bodmer and Lukas Vogel and René Zurbrügg and David Helm and Rahel Rickenbach and Simon Muntwiler and Jerome Sieber and Melanie N. Zeilinger., "Chronos and CRS: Design of a miniature car-like robot and a software framework for single and multi-agent robotics and control " arXiv preprint arXiv:2209.12048, 2022.