International Journal of Innovative Technology and Interdisciplinary Sciences
www.IJITIS.org
ISSN:2613-7305
Volume 6, Issue 1, pp. 1081-1099, 2023
DOI: https://doi.org/10.15157/IJITIS.2023.6.1.1081-1099
Received November 11, 2022; Accepted December 30, 2022

Autonomous Driving Vehicles and Control System Design

Trieu Minh Vu*[a1], Reza Moezzi[b1], Klodian Dhoska[c2]

[1] Institute for Nanomaterials, Advanced Technologies and Innovation, Technical University of Liberec, Czechia

[2] Faculty of Mechatronics, Informatics and Interdisciplinary Studies, Technical University of Liberec, Czechia

[2] Department of Production and Management, Faculty of Mechanical Engineering, Polytechnic University of Tirana, Albania

*[a]trieu.minh.vu@tul.cz; [b]Reza.Moezzi@tul.cz; [c]kdhoska@fim.edu.al

ABSTRACT

Autonomous driving vehicles and the control system design have been undergoing rapid changes in the last decade and affecting the concept and behaviour of human traffic. However, the control system design for autonomous driving vehicles is still a great challenge since the real vehicles are subject to enormous dynamic constraints depending on the vehicle physical limitations, environmental constraints and surrounding obstacles. This paper presents a new scheme of nonlinear model predictive control subject to softened constraints for autonomous driving vehicles. When some vehicle dynamic limitations can be converted to softened constraints, the model predictive control optimizer can be easier to find out the optimal control action. This helps to improve the system stability and the application for further intelligent control in the future. Simulation results show that the new controller can drive the vehicle tracking well on different trajectories amid dynamic constraints on states, outputs and inputs.

Keywords: Autonomous Driving Vehicles, Control Optimizer, Hard Constraints, Softened Constraints, Control System Design, Model Predictive Control.

1. INTRODUCTION

Autonomous driving vehicles and the control system design have undergone rapid and widespread development in the last decade. Applications of advanced control techniques, AI and communication networks have been making autonomous driving vehicles changing our society. Controllers for autonomous driving vehicles and driver assistance systems can be based on model-free or model-based models. For model-free controllers, the feedback error and the control action are mostly generated from fuzzy logic, neural networks and AI. However, due to the limit size of this paper, we mainly focus on the model-based controllers using objective functions subject to dynamic constraints. This paper mainly uses the vehicle dynamic modelling from the book reference in [1].

Controller design for autonomous driving vehicles can be implemented with conventional PID, H_2 and H_∞ feedback controllers. An adaptive PID controller with integration of multi-sensor navigation and trajectory tracking is presented in [2]; The

controller can maintain the system stability and handle the instantaneous trajectory error. A new conventional controller based on sliding mode and fuzzy logic is introduced in [3]; The authors presented a new robust adaptive controller for trajectory tracking and lane keeping for autonomous driving vehicles dealing with unstructured uncertainties and disturbances. Path-tracking issue for controlling autonomous driving vehicles with integral sliding mode is also presented in [4]; The authors designed the radial basis function neural network and nonlinear feedback-based integral sliding mode controller and extended Kalman filter to ensure the system stability and minimize the tracking error.

A new conventional method of active disturbance rejection control based on PID, nonlinear feedback, and robust control based on the extension of the vehicle model is referred to [5]; This method deals with the velocity varying and the lateral uncertain disturbances, feedforward and feedback to control the lateral and longitudinal motion. A brief comparison among trajectory tracking controllers for autonomous driving vehicles is presented in [6]; This paper summarized conventional control methods of H_2, H_∞, PID, sliding mode and linear quadratic regulator (LQR). Authors in [7] showed that, the LQR strategies are more suitable and robust against the system uncertainties and measurement noises. Model Predictive Control (MPC) develops from the LQG algorithms. LQG refers to infinite horizon optimizer and the algorithms are much simpler and deal better with disturbance rejection; While MPC refers to finite horizon optimizer, the algorithms are more complex and needed to perform more complicated calculations online. However, MPC shows better trajectory tracking and considerably smoother control action changes [8].

LQR and linear matrix inequalities (LMI) are widely applied for online control optimizer subject to dynamic constraints that the conventional controllers cannot be used. A new method for autonomous driving vehicle with Lyapunov function based on LQR-LMI algorithms is proposed in [9]; In this model-based controller LQR and MPC, the system is considered fully modelled and all states are considered fully observed. The mismatch between the model and the real vehicle as well as the noises and uncertainties from the system are not considered. Therefore, several linear quadratic Gaussian (LQG) methods are studied and applied with Gaussian noises as well as the uncertain measured outputs, where the full system states may not need to be observed. The design of LQG with adaptive Q-matrix improves the vehicle tracking performance in [10]; This method can handle better the model-plant mismatch and noises.

Recent research references of model-based MPC methods for controlling autonomous vehicles are enormous. Some highlighting recent MPC references are found in [11-21]: A new presentation for robust MPC (RMPC) subject to the uncertain system using LMIs subject to inputs and outputs saturated constraints is presented in [11]; This paper describes a new RMPC method with polytopic uncertainties and constraints for linear time varying (LTV); This RMPC can maintain the system stability amid the presence of the system uncertainties. [12] presents the generation of feasible paths for autonomous mobile robots and nonlinear model predictive control (NMPC); Several NMPC methods are developed and compared to show the ability of the NMPC to maintain the system stability and trajectory tracking ability.

At [13] presents a controller for autonomous vehicle steering system in MIMO system; A new adaptive MPC (AMPC) is implemented. This AMPC provides better trajectory tracking performances for dynamic changing systems. [14] develops a new

fault tolerant AMPC algorithm of robust trajectory tracking control for autonomous vehicles; This method includes AMPC and novel Kalman filter; The proposed method can detect and isolate the faults and keep the system stability. At [15] presents the implementation of MPC-based trajectory tracking with hard constraints on outputs and inputs; In this method, the nonlinear model is linearized and discretized. A review of shared control for automated vehicles for advanced driver assistance systems (ADAS) presents in [16]; Most of the controllers recently used in autonomous systems are LQ, H_2, H_∞, LMI- H_∞, LMI-LQ, and MPC; However, there is still no attempt to develop MPC with softened constraints. [17] introduces a novel method of MPC-based with PID for control of autonomous vehicles tracking trajectories; The PID provides feedback from the MPC optimizer; Simulations show good performances of trajectory tracking and speed tracking. [19] presents a new MPC proposal for autonomous driving vehicles; MPC optimizer calculates the optimal inputs of steering wheel and vehicle speed subject to the vehicle physical constraints. And finally, [20, 21] present novel control algorithms based on fuzzy control for intelligent vehicle lane change and tracking setpoints for RMPC. Latest advanced control system designs are referred to from reverence [22-29].

From the recent reference reviews, there is still lack of MPC application with softened constraints. The idea of this paper comes from the fact that, MPC is a finite horizon optimizer subject to dynamic constraints. If we include all these constraints into this optimizer, the MPC will have a lot of constraints on states, inputs and outputs, and therefore, the optimizer may not find out a solution. Since the MPC is designed for an on-line calculation and any infeasible solution is not tolerated. Thus, it would be better if we converse some constraints from the vehicle dynamics into softened constraints, it will widen the ability of the MPC optimizer to find out solution and will improve considerably the stability and robustness of this MPC controller.

The structure of this paper is as follows: Part 2 introduces the vehicle modelling and constraints; Part 3 presents the NMPC with hard constraints; Part 4 presents the new NMPC with softened constraints; Part 5 illustrates the two schemes' performances; and finally, Part 6 is our conclusion and recommendation.

2. VEHICLE MODELLING AND CONSTRAINTS

In this paper, the vehicle model is based on [1], where the vehicle is modelled as a four wheels model. This vehicle is assumed to be totally identified on x and y coordinate at the centre of the rear wheels by the Global Positioning System (GPS). The vehicle body angle, θ, and the steering angle, φ, are also always identified by 3D sensors system embedded into the vehicle. The distance between the centre of the front wheels and the rear wheels is called the vehicle wheelbase, l, and the rolling radius of the vehicle wheel is r, as shown in Figure 1.

The vehicle steering wheel can rotate in a hard limit angle of +/- 675 degrees and make the front wheels turning in hard angle ranges of +/- 45 degrees. From now on, for the simplicity purpose, we call the vehicle steering wheel angle as the vehicle front wheels angle because this angle is used to calculate the vehicle movement direction. In all calculations and simulations from now on, we use the vehicle wheelbase, $l = 2$ meters, and the wheel rolling radius, $r = 0.25$ metre. It is assumed that all of the vehicle parameters are totally identified and always measured by x, y, θ, and φ. Therefore, the

vehicle can be totally controlled by the two inputs, $\dot{\theta}$, and $\dot{\phi}$ (the angular velocity of the vehicle driven rolling wheels and the steering wheel).

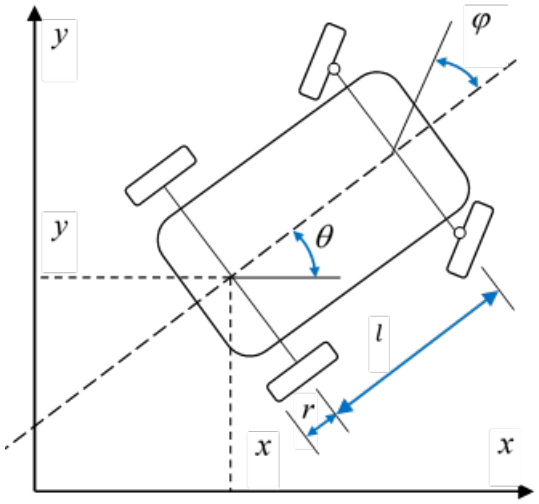

Figure 1. Vehicle modelling

The vehicle dynamics in [1] shows that the vehicle can move forward and reverse as well can be driven by front wheels or by rear wheels. Equation (1) shows the vehicle moving forward and driven by the rear wheels

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ \frac{\tan\phi}{l} \\ 0 \end{bmatrix} r v_1 + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} v_2 \tag{1}$$

Equation (2) shows the vehicle driven by the rear wheels but moving in reverse speed. In the calculations and simulations, we assign the reverse speed as in negative sign.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = -\begin{bmatrix} \cos\theta \\ \sin\theta \\ \frac{\tan\phi}{l} \\ 0 \end{bmatrix} r v_1 + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} v_2 \tag{2}$$

Finally, equation (3) shows the vehicle driven by front wheels and moving forwards, where we control the vehicle speed from the front rolling wheels.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \cos\theta\cos\phi \\ \sin\theta\cos\phi \\ \frac{\tan\phi}{l} \\ 0 \end{bmatrix} r v_1 + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} v_2 \tag{3}$$

In the above equations, $[x, y, \theta, \phi]'$ are the vehicle states and outputs. The two control inputs are the vehicle rolling wheel angular velocity, v_1, and the steering wheel angular velocity, v_2. Therefore, rv_1 is the vehicle speed in kilometre per hour (km/h), and v_2 is the vehicle steering angular velocity in revolutions per minute (rpm).

A real vehicle always has a strict steering angle limit at:

$$-\frac{\pi}{4} \leq \phi \leq \frac{\pi}{4} \tag{4}$$

The vehicle represented in equations (1) or (2) or (3) can be controlled for tracking a given trajectory from a given starting position to a given destination position. It is assumed that the vehicle must start from an initial position $[x_0, y_0, \theta_0, \phi_0]$ at the time $t = 0$, and moving to the destination at the end of a trajectory, $[x_T, y_T, \theta_T, \phi_T]$ at the time $t = T$.

In vehicle dynamics, the vehicle tire slip will be taken place when the vehicle speed is greater than 12 km/h. Then, the vehicle sideslip will increase exponentially when the vehicle speed exceeds 67 km/h. Therefore, the vehicle steering angle must be always put in strict constraints of the vehicle speed, as shown in Figure 2.

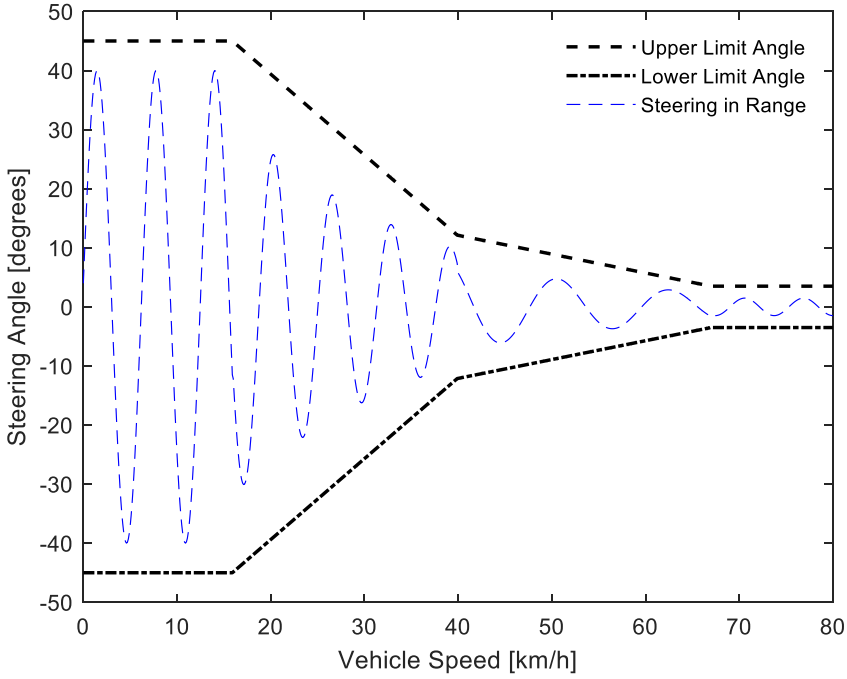

Figure 2. Vehicle steering angle vs vehicle speed

Figure 2 shows that, at low speed of less than 16 km/h, the steering angle can move almost freely in its limit of +/- 45 degrees. The steering angle will be reduced to less than +/- 12 degrees when the vehicle speed increases from 16 km/h to 40 km/h. Then, the limit of steering angle will rapidly reduce to less than +/- 4 degrees as the vehicle speed exceeds more than 67 km/h. These dynamic constraints can be changed somehow into softened constraints in order to widen the ability of the MPC controller to find out the solution. The basic MPC algorithms will be presented in the next part

3. NMPC WITH HARD CONSTRAINTS

Vehicle models in (1), (2), and (3) are all nonlinear forms and can be linearized and transformed into discretized time models. Equations (1), (2), and (3) can be considered as the first order continuous derivative equation as:

$$\dot{X} = f(x, u) \tag{5}$$

where x is the state variables, $x \triangleq [x, y, \theta, \phi]'$, and u is the inputs, $u = [u_1, u_2]'$. The first order nonlinear in (5) can be approximated in a Taylor series at any referenced position of (x_r, u_r) for $\dot{X}_r = f(x_r, u_r)$, that:

$$\dot{X} \approx f(x_r, u_r) + f_{x,r}(x - x_r) + f_{u,r}(u - u_r) \tag{6}$$

in which, $f_{x.r}$ and $f_{r.x}$ are the Jacobean function of x and u, moving around the referenced positions (x_r, u_r).

Subtraction (6) for $\dot{X}_r = f(x_r, u_r)$, we can obtain an approximation linear form for the continuous time (t):

$$\dot{\tilde{X}}(t) = A(t)\tilde{X}(t) + B(t)\tilde{u}(t) \tag{7}$$

in which the approximation of $\tilde{X}(t) = X(t) - X_r(t) = \begin{bmatrix} x(t) - x_r(t) \\ y(t) - y_r(t) \\ \theta(t) - \theta_r(t) \\ \phi(t) - \phi_r(t) \end{bmatrix}$, and $\tilde{u}(t) = u(t) - u_r(t) = \begin{bmatrix} u_1(t) - u_{r1}(t) \\ u_2(t) - u_{r2}(t) \end{bmatrix}$,

$$A(t) = \begin{bmatrix} 0 & 0 & -u_{r1}(t)\sin\theta_r(t) & 0 \\ 0 & 0 & u_{r1}(t)\cos\theta_r(t) & 0 \\ 0 & 0 & 0 & \frac{u_{r1}(t)}{l\cos^2\phi_r(t)} \\ 0 & 0 & 0 & 0 \end{bmatrix}, B(t) = \begin{bmatrix} \cos\theta_r(t) & 0 \\ \sin\theta_r(t) & 0 \\ \frac{\tan\phi_r(t)}{l} & 0 \\ 0 & 1 \end{bmatrix}$$

The continuous approximation form of $\dot{\tilde{X}}(t)$ in equation (7) can be transferred into the discrete-time form in k and $k + 1 = k + \Delta t$, with Δt being the length of the sampling interval or the computer scanning speed. The discrete inputs $u(k)$ will be kept at constant values from the time interval k to $k + 1$. The discrete form for NMPC optimizer now can be written as:

$$\begin{aligned} \tilde{X}(k+1) &= A(k)\tilde{X}(k) + B(k)\tilde{u}(k) \\ \tilde{Y}(k) &= C(k)\tilde{X}(k) \end{aligned} \tag{8}$$

in which,

$$A(k) = \begin{bmatrix} 1 & 0 & -u_{r1}(k)\sin\theta_r(k)(\Delta t) & 0 \\ 0 & 1 & u_{r1}(k)\cos\theta_r(k)(\Delta t) & 0 \\ 0 & 0 & 1 & \frac{u_{r1}(k)}{l\cos^2\phi_r(k)}(\Delta t) \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$B(k) = \begin{bmatrix} \cos\theta_r(k)(\Delta t) & 0 \\ \sin\theta_r(k)(\Delta t) & 0 \\ \frac{\tan\phi_r(k)}{l}(\Delta t) & 0 \\ 0 & (\Delta t) \end{bmatrix},$$

$$C(k) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

and,

$$\tilde{X}(k) = X(k) - X_r(k) = \begin{bmatrix} x(k) - x_r(k) \\ y(k) - y_r(k) \\ \theta(k) - \theta_r(k) \\ \phi(k) - \phi_r(k) \end{bmatrix}, \text{ and}$$

$$\tilde{u}(k) = u(k) - u_r(k) = \begin{bmatrix} u_1(k) - u_{r1}(k) \\ u_2(k) - u_{r2}(k) \end{bmatrix}$$

In those approximation discretized vehicle dynamics, there are two control inputs of vehicle speed, $u_1(k) - u_{r1}(k)$, and the steering angular velocity, $u_2(k) - u_{r2}(k)$. There are four measured outputs, $y(k) = \tilde{Y}(k) = C(k)\tilde{X}(k)$. These outputs will be updated at each time interval. The discretized vehicle dynamics in (8) is the time variant model since this system is depending on the time interval updating or the computer scanning speed, Δt.

The MPC algorithms for this vehicle dynamics can be expressed in finite horizon prediction outputs and inputs. For the simplicity, from now on, we assign the horizon output equal to the horizon input or $N_u = N_y$. The MPC objective function for the vehicle tracking trajectory subject to hard constraints will be:

$$\min_{U \triangleq \{\Delta u_k, \ldots, \Delta u_{k+N_u-1}\}} \Big\{ J\big(U, x(k)\big) = \sum_{i=0}^{N_y-1} \big[(y_{k+i|k} - r_{k+i|k})' Q (y_{k+i|k} - r_{k+i|k}) + \Delta u'_{k+i|k} R \Delta u_{k+i|k} \big] \Big\}, \tag{9}$$

subject to:

$u_k \in \mathcal{U}$, and $u_{k+i} \in [umax_{min}]$, $\Delta u_{k+i} \in [\Delta umax_{min}]$, for $i = 0,1, \ldots, N_u - 1$,

$y_k \in \mathcal{Y}$, and $y_{k+i|k} \in [ymax_{min}]$, for $i = 0,1, \ldots, N_y - 1$,

$\Delta u_k = u_k - u_{k-1} \in \Delta\mathcal{U}$, and $\Delta u_{k+i} = 0$, for $i \geq N_u$,

$x_{k|k} = x(k)$, $x_{k+i+1|k} = A(k)x_{k+i|k} + B(k)u_{k+i}$, $u_{k+i|k} = u_{k+i-1|k} + \Delta u_{k+i|k}$, $y_{k+i|k} = C(k)x_{k+i|k}$,

where $x(k)$ are the state variables at the present discrete time (k), $U \triangleq \{\Delta u_k, \ldots, \Delta u_{k+N_{u-1}}\}$ is the solution of predictive input horizon from k to N_u. And N_y is the predictive output horizon; $y_{k+i|k}$ are the outputs at the present discrete time (k), $r_{k+i|k}$ is the tracking trajectory setpoints; $\Delta u_{k+i|k}$ is the input predictive increments, $\Delta u_{k+i|k} = u_{k+i|k} - u_{k+i-1|k}$; $Q = Q' \geq 0$, $R = R' > 0$ are the weighting matrices for outputs and inputs, respectively.

By substituting $x_{k+j|k} = A^k x(k) + \sum_{j=0}^{k-1} A^j B u_{k+j-1-j}$, equation (9) can be rewritten as

$$V\big(x(k)\big) = \frac{1}{2}x'(k)Yx(k) + \min_{U}\left\{\frac{1}{2}U'HU + x'(k)FU\right\} \tag{10}$$

subject to the linear matrices v inequality (LMI), $GU \leq W + Ex(t)$, where the column vector $U \triangleq \left[u_k^{'}, \dots, u_{k+N-1}^{'}\right] \in \mathbb{R}^s$, $s \triangleq mN_u$ is the optimization vector, $H = H^{'} > 0$, and *H, F, Y, G, W, E* are obtained from *Q, R* and in (9) as only the optimizer vector *U* is needed, the term involving *Y* is usually removed from (10). The optimization problem (10) is a quadratic program (QP). The MPC optimizer will calculate the optimal input vector $U \triangleq \{\Delta u_k, \dots, \Delta u_{k+N_{u-1}}\}$ subject to the hard constraints of the inputs, $u_k \in \mathcal{U}$, and $u_{k+i} \in [umax_{min}]$; of the outputs $y_k \in \mathcal{Y}$, and $y_{k+i|k} \in [ymax_{min}]$; and of the input increments $\Delta u_{k+i} \in [\Delta umax_{min}]$. But only the first input increment, Δu_k, is taken into the implementation. Then, the optimizer will update the outputs and states variables with the new update input and repeat the calculation for the next time interval. Therefore, the MPC is also called as the receding time horizon control. A diagram control system for this NMPC is shown in Figure 3.

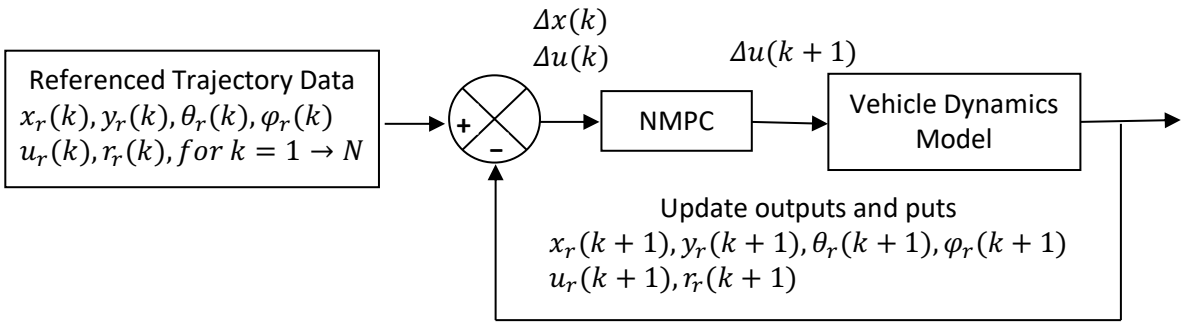

Figure 3. NMPC diagram system

The NMPC optimizer in Figure 3 receives the online optimal control action, $\Delta u(k)$, and feeds into the vehicle dynamics model and update the update current states, inputs and outputs. The update states, inputs and outputs will feedback and compare to the reference trajectory data. The new differences of states, inputs and outputs, then, again feed in the NMPC optimizer for the next online optimal input $\Delta u(k)$ calculation.

Next part, we present the development of NMPC with softened constraints.

4. NMPC WITH SOFTENED CONSTRAINTS

When all constraints are set into hard constraints, difficulty will arise since the controller may not find out the solution satisfying all constraints and the controller may become infeasible.

In reality, some physical constraints can be violated a little bit during evolution of the system since some initial conditions may lead to some violations in constraints. So that we can consider and assign some constraints as softened constraints in order to widen the possibility of the MPC to find out optimal solution. The softened constraints can be formulated into the following form:

$$\begin{bmatrix} 1 & z_i' \\ z_i & X + \mu\varepsilon_i I \end{bmatrix} \geq 0 \tag{11}$$

$$\begin{cases} \min_j X_{jj} \le x_{max}^2 \\ \forall z_i \in vert\left\{\chi_{u^*(.|k)}^{k+i|k}\left(x(k)\right)\right\}, \forall i \in \{1,..,N\} \end{cases}$$

where μ is assigned as big values as a weighting factor ($\mu > 0$), and ε_i is the constraints penalty terms ($\varepsilon_i \ge 0$) added into the MPC objective function. X and z_i are the corresponding matrix of the hard constraints. So that some hard constraints can be converted into the softened form. The new MPC algorithm subject to softened constraints can be written as:

$$\min_{U \triangleq \{\Delta u_k, \dots, \Delta u_{k+N_u-1}\}} \Bigg\{ J(U, x(k) = \sum_{i=0}^{N_y-1} \left[(y_{k+i|k} - r_{k+i|k})' Q \left(y_{k+i|k} - r_{k+i|k}\right) + \Delta u'_{k+i|k} R \Delta u_{k+i|k} + \varepsilon'_i(k) \Lambda \varepsilon_i(k) + 2\mu' \varepsilon_{k+i|k} \right] \Bigg\} \quad (12)$$

subject to (11) and:

$u_k \in \mathcal{U}$, and $u_{k+i} \in [umax_{min}]$, $\Delta u_{k+i} \in [\Delta umax_{min}]$, for$i = 0,1, \dots, N_u - 1$,

$y_k \in \mathcal{Y}$, and $y_{k+i|k} \in [ymax_{min}]$, for$i = 0,1, \dots, N_y - 1$,

$\Delta u_k = u_k - u_{k-1} \in \Delta\mathcal{U}$, and $\Delta u_{k+i} = 0$, for $i \ge N_u$,

$x_{k|k} = x(k)$, $x_{k+i+1|k} = A(k)x_{k+i|k} + B(k)u_{k+i}$, $u_{k+i|k} = u_{k+i-1|k} +$

$\Delta u_{k+i|k}$,$y_{k+i|k} = C(k)x_{k+i|k}$,

where, $\varepsilon_i(k) = \left[\varepsilon_y; \varepsilon_u\right]$, $yy_{k+i|k} y_{max_{min}}$and $uu_{k+i|k} u_{max_{min}}$; And $\Lambda = \Lambda' \ge 0$ is the additional penalty matrix (generally $\Lambda > 0$ and assign to small values); In this new NMPC, the penalty term of soften state constraints $\sum_{i=0}^{N_p}\left[\varepsilon'_{k+i|k} \Lambda \varepsilon_{k+i|k} + 2\mu' \varepsilon_{k+i|k}\right]$ is added into the objective function with positive definite and symmetric matrix Λ; This term penalizes violations of softened constraints and when possible, the free constrained solution will be returned.

Now this NMPC calculates the new optimization vector $U_S = \begin{bmatrix} U \\ \varepsilon \end{bmatrix}$ and the new NMPC computational algorithms will be:

$$\Psi_S\left(x(t)\right) = \min_{U_S} \left\{ \frac{1}{2} U_S' H_S U_S + x'(t) F_S U_S \right\}, \quad (13)$$

subject to $G_S U_S \le W_S + E_S x(k)$,

where $U_S \triangleq \left[u_k^{'}, u_{k+1}^{'}, \ldots, u_{k+N_p-1}^{'}, \varepsilon_k^{'}, \varepsilon_{k+1}^{'}, \ldots, \varepsilon_{k+N_p}^{'}\right]^{'}$ is the optimization vector, $H_S = \begin{bmatrix} H & 0 \\ 0 & M \end{bmatrix}$ and $F_S = [F \quad \mu]$, and matrices for inequality constraints *H, F, G, W,* and *E* are obtained from equation (10),

$$G_S = \begin{bmatrix} G & 0 \\ g_S & -I \\ 0 & -I \end{bmatrix} \text{ with } g_S = \begin{bmatrix} 0 & 0 & 0 & \ldots & 0 \\ ZB & 0 & 0 & \ldots & 0 \\ ZAB & ZB & 0 & \ldots & 0 \\ \ldots & \ddots & \ddots & \ddots & \vdots \\ ZA^{N_p-1}B & ZA^{N_p-2}B & \ldots & \ldots & ZB \end{bmatrix},$$

$$W_S = \begin{bmatrix} W \\ w_S \\ 0 \end{bmatrix} \text{ with } w_S = \begin{bmatrix} z \\ \vdots \\ z \end{bmatrix}, \text{ and } E_S = \begin{bmatrix} E \\ e_S \\ 0 \end{bmatrix} \text{ with } e_S = \begin{bmatrix} -Z \\ -ZA \\ -ZA^2 \\ \vdots \\ -ZA^{N_p} \end{bmatrix}.$$

To illustrate the ability of the new controller, we test the two NMPC schemes in (9) and in (12) with following simple example as considering the below nonlinear system:

$$\begin{aligned} \dot{x}_1 &= 2x_2 + u(1 + x_1) \\ \dot{x}_2 &= 2x_1 + u(1 - 3x_2) \end{aligned} \tag{14}$$

It is assumed that this system in (14) is subjected to the hard state and input constraints $x_{min} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$ and $-2 \le u \le 2$. The linearized approximation of this system from (7) is: $\dot{x} = Ax + Bu$, in which, $A = \begin{bmatrix} 0 & 2 \\ 2 & 0 \end{bmatrix}$ and $B = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$. The weighting matrices are chosen as $Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ and $R = 1$. The weighting matrices for softened constraints are chosen as $\Lambda = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ and $\mu = 10{,}000$. It is assumed that the system is starting form an initial state position, $x_0 = \begin{bmatrix} -0.72 \\ -0.35 \end{bmatrix}$. Figure 4 shows the performances of two NMPC schemes: This initial state position x_0 does not lead to any violation of states and input ($x_{min} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$ and $-2 \le u \le 2$). In this x_0, the solutions of the two control schemes are always available. We can see that, the NMPC with softened state approaches the asymptotic point faster than the hard constraints. It means that, if we loosen somehow some constraints, the optimizer can generate easier optimal inputs and the system will be more stable.

Now, it is interesting to see in Figure 4 that, both schemes have $x_{1\,min}^{Hard} = -0.8475$ and $x_{1\,min}^{Softened} = -0.8483$, almost reach the hard constraint of $x_{min} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$. These states still have not violated the state constraints but if we select some other initial positions x_0, that may lead to some state and input violations.

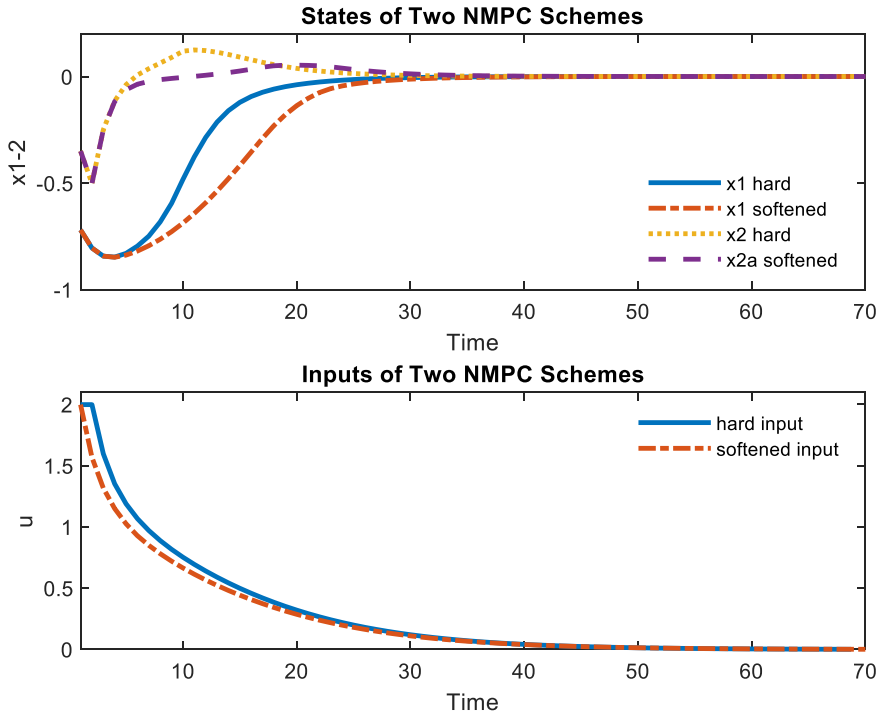

Figure 4. Comparison of two RMPC schemes

Now, if we select $x_0 = \begin{bmatrix} -0.9 \\ -0.8 \end{bmatrix}$, this initial condition will lead to the violations of the state and the input constraints as $x_{1\,min} = -1.0441$ and $u_{max} = 2.2303$. These violations make the RMPC with hard constraints infeasible. Meanwhile, the RMPC scheme with softened constraints is still working well and still easily to find out optimal input solutions as shown in Figure 5. And after a short transitional period, the fully constrained solution is returned or there is no more constrained violation.

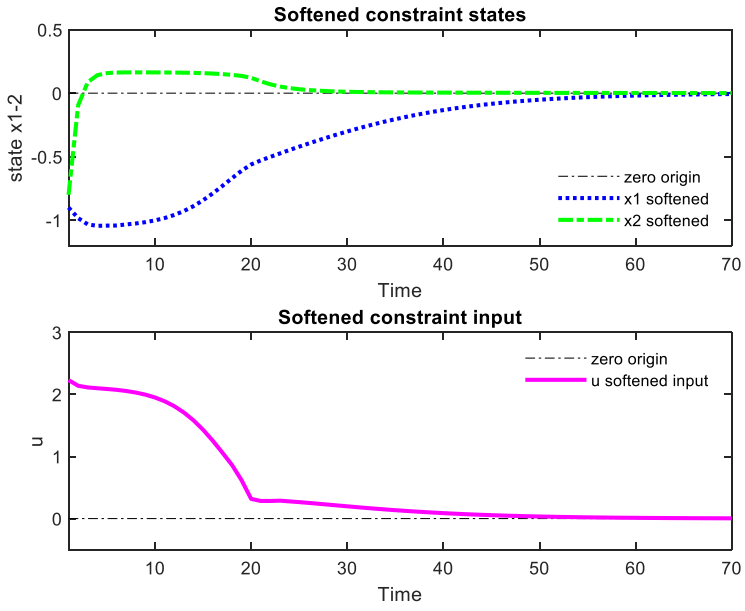

Figure 5. Softened Constraint RMPC

The two NMPC schemes will be further analysed and simulated in the next part with different trajectories and control parameters.

5. NMPC TRACKING TRAJECTORY PERFORMANCES

Firstly, we test the two NMPC schemes tracking on a full circle from an initial position outside. In this example we select an initial position of $x_0 = [-0.5 \quad -0.5 \quad 0 \quad 0]'$. The constraints are imposed on this vehicle as: the input limits, $u[-1,-1]'_{min}$, $u[1,1]'_{max}$; the increasement of input limits, $\Delta u[-0.5,-0.5]'_{min}$, $\Delta u[0.5,0.5]'_{max}$; and the coordinate limits, $y[-1,-1,-1,-1]'_{min}$, and $y[1,1,1,1]'_{max}$; In our NMPC algorithms, the predictive horizons are set with $N_u = N_y = 10$; The state and the input penalty matrices are set with $Q = diag\{1,1,1,1\}$ and $R = diag\{1,1\}$. Performances of the two MPC schemes are shown in Figure 6.

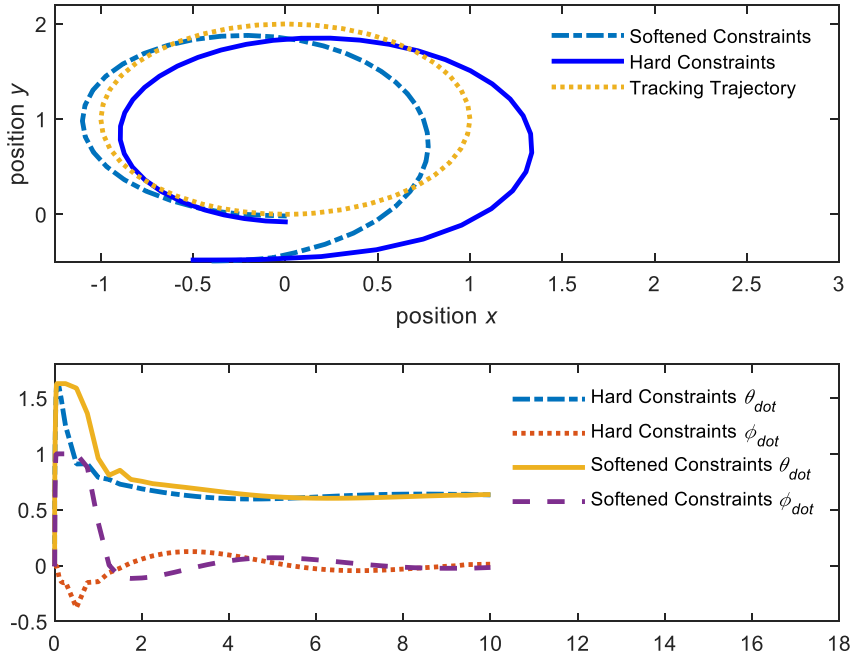

Figure 6. NMPC schemes tracking a circle

Figure 6 shows that, the softened constraints scheme goes to setpoints faster than the hard constraints. The control input actions of the softened constraints are also likely smoother than hard constraints. However, the softened constraints scheme is more complex and leads to longer elapsed CPU time (0.89 sec) vs hard constraints scheme (0.74 sec).

Next, we test these two schemes tracking on real polynomial trajectories with different MPC control parameters to have a look closer inside the ability of each scheme. Now, we assume having a feasible polynomial trajectory from x_0, y_0 of [0, 0] to x_T, y_T of [10, 10]. The vehicle is starting from an initial condition at $[x_0, y_0, \theta_0, \phi_0] = [0, -0.5, 0, 0]'$, and arriving the destination condition at $[x_T, y_T, \theta_T, \phi_T] = [10, 10, 0, 0]'$ The prediction horizon is set with $N_u = N_y = 10$; The penalty matrices for states and

inputs are set with $Q = diag\{1, 1, 1, 1\}$ and $R = diag\{1, 1\}$; The vehicle speed vs the steering angular velocity are fully controlled. Performances of the two schemes are shown in Figure 7.

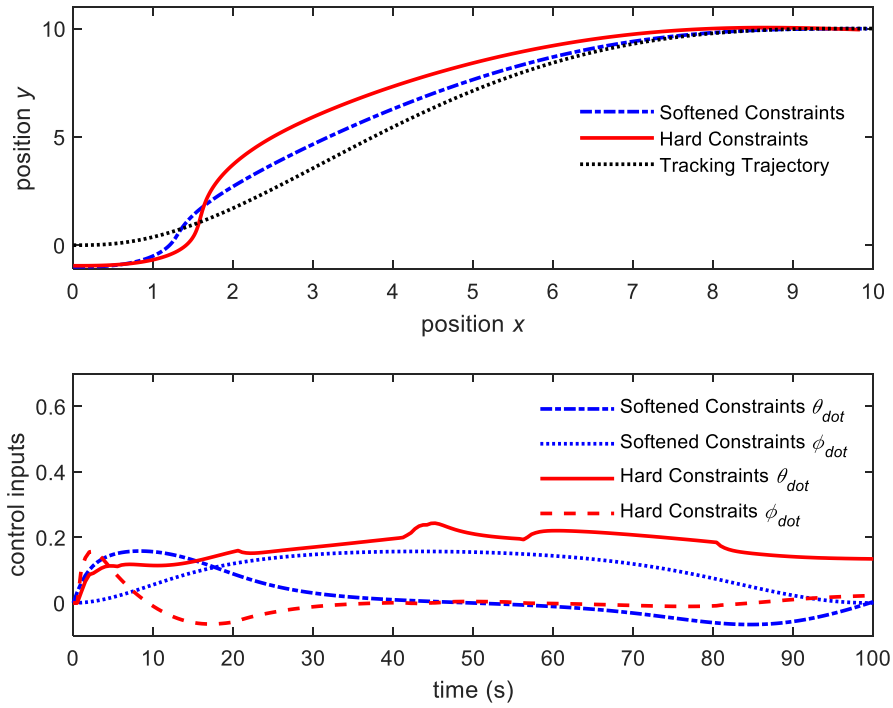

Figure 7. NMPC schemes tracking trajectory

Figure 7 shows that the softened constrained scheme goes faster to track on the trajectory and always maintains the tracking errors smaller than the hard-constrained scheme. The control inputs of the softened scheme are also smoother. The hard-constrained scheme becomes more difficult to drive the vehicle tracking to the trajectory. However, the CPU elapsed time of the softened scheme becomes great challenge for computer system. The time of elapsed CPU for softened constraints scheme is 4.27 secs, while the time for hard constrained scheme is only 2.45 secs for the whole trajectory control.

In order to shorten the CPU elapsed time, we try to reduce the MPC prediction horizon. But the too short prediction horizon will lead to the harder control actions and will lead the system to infeasible and instable. Figure 8 shows the performances of the two schemes with shorter state and input prediction horizon of $N_u = N_y = 5$.

When we shorten the state and control prediction horizon to *Ny*=*Nu*=5, both schemes are still stable and working well. But the hard-constrained scheme likely generates harder control actions and has more difficult to approach to the trajectory. However, the hard-constrained scheme needs only 1.84 secs for elapsed CPU time while the softened scheme consumes of 3.23 secs for elapsed CPU time for the whole drive.

If we lengthen the prediction horizon, the system will become loose, more flexible but it will lengthen considerably the CPU time, and have bigger tracking errors. Figure 9 shows the performances of the softened constrained scheme with the control horizon of *Ny*=*Nu*=10 vs *Ny*=*Nu*=30.

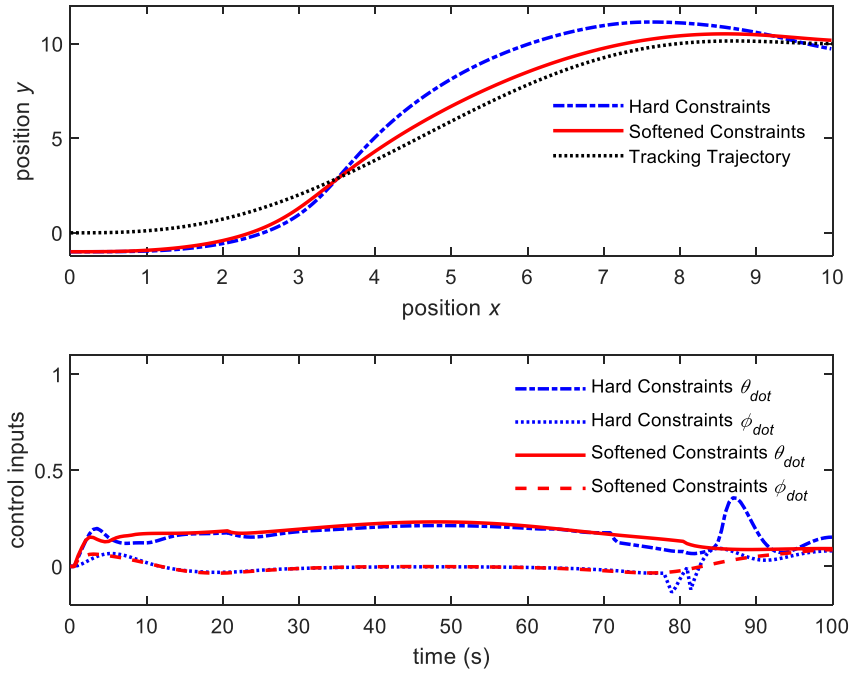

Figure 8. NMPC schemes with shorten prediction horizon N=5

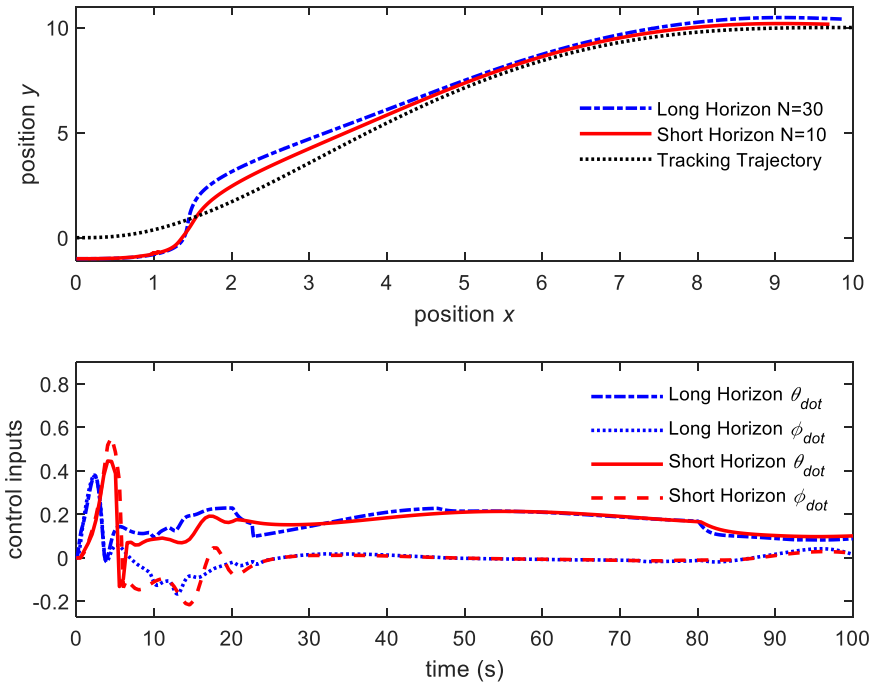

Figure 9. NMPC schemes with N=10 vs N=30

The short control prediction horizon leads to the harder control actions and the system approaches the setpoint faster. However, the longer horizon leads to smoother control actions and system becomes loose and more stable. The time for elapsed CPU for long prediction horizon, N=30, is 7.65 secs, considerably greater than the time for CPU with short prediction horizon N=10, of 4.47 secs.

In MPC algorithms, the CPU discrete time interval or computer scanning speed is also an important factor affecting their performances. The MPC discretized system is a time variant model and depending on the length of time intervals or the computer scanning speed. Figure 10 shows the performance of the two schemes with scanning time interval of 0.1 sec vs 0.5 sec.

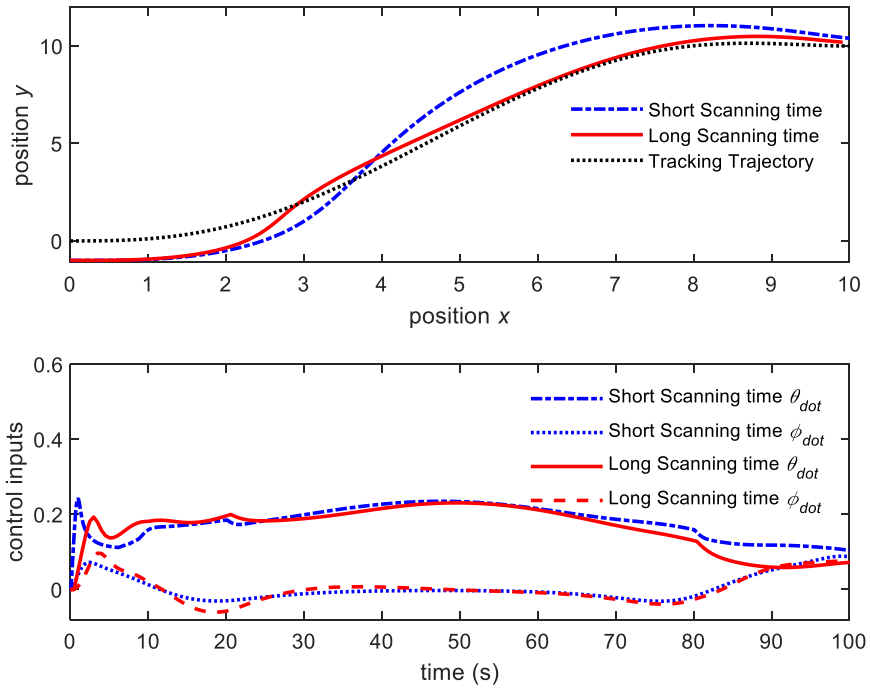

Figure 10. NMPC schemes with short vs long time interval

The too fast scanning speed will lead to instable. The control system will become more sensitive and more difficult to control. When the scanning time interval is set at 0.1 sec, the time of elapsed CPU will be 21.74 secs. While the scanning time interval is set at slower speed of 0.5 sec, the CPU elapsed time reduces to only 4.15 secs.

Finally, we illustrate the run of different state and input penalty matrices, Q and R. If we set R too bigger than Q, it means that a small change of input will lead to a big value at the objective function. The control system becomes less sensitive and more stable. But it becomes more difficult to track the setpoints. On the other hand, if we set R too small values to Q, the control actions will become harder, and the system will approach the setpoints faster. But the system will become less stable. Figure 11 shows the performances of the softened constrained scheme with R=60 and R=1.

If we set the input penalty matrix, R=60, the control system becomes less sensitive to any change of the inputs since the inputs can be changed in only small increasements (Light Input Changes). The system goes smoother and more stable. But the tracking errors become bigger. If we set the input penalty matrix, R=1 only, the inputs can change harder (Heavy Input Changes), and the tracking errors are smaller. But the control system will become less stable.

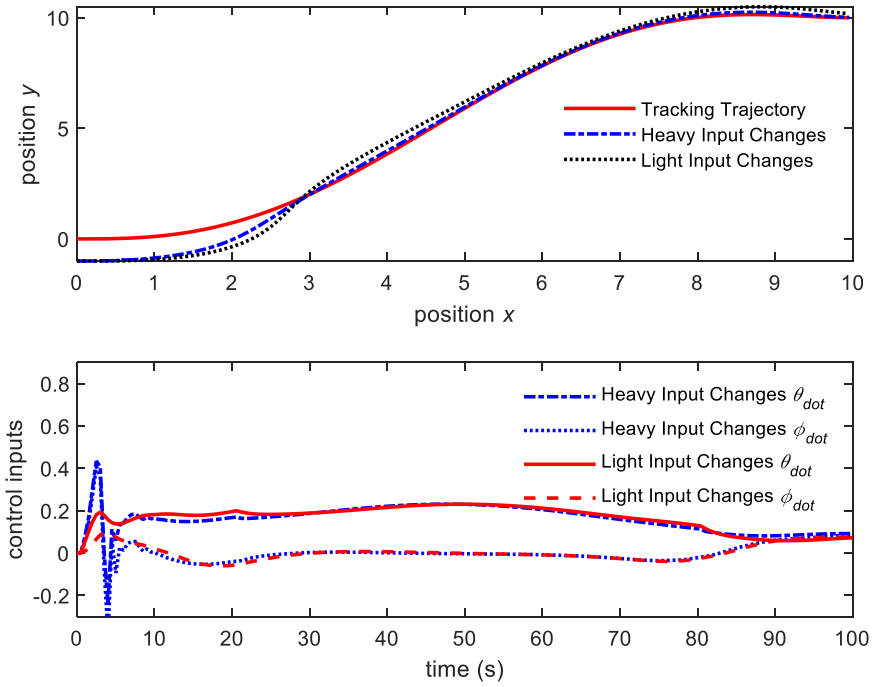

Figure 11. NMPC with light vs heavy input matrix R=60 vs R=1

The above simulations of both NMPC schemes for tracking different trajectories with different control parameters show that the NMPC scheme with softened constraints is more stable but really needs more time of elapsed CPU and more complicated and complex in programming.

6. CONCLUSION

The new NMPC subject to softened constraints has shown its ability to maintain the stability amid presence of enormous constraints on states, outputs and inputs where the conventional NMPC schemes with hard constraints become infeasible because of the constrained violations. Conversion of some hard constraints into softened constraints helps to widen the feasible boundary for optimal control actions. The constrained violations are usually taken place in short transitional periods until the NMPC optimizer finding out the optimal control actions, that fully satisfy all constraints. The new controller scheme can be also applied for intelligent control of neural networks, fuzzy logic and AI in the future. The next research will be focused on the real vehicles and the controller will be the combination of advanced control techniques including online video processing, GPS, LIDAR, and high internet human-machine interfaces.

ACKNOWLEDGEMENT

The authors would like to thank Technical University of Liberec, Czech Republic, for helpful suggestions and supporting to submit this paper. The result was obtained through financial support from Ministry of Education, Youth and Sports in Czechia and the European Union in the framework of the project “Modular platform for autonomous

chassis of specialized electric vehicles for freight and equipment transportation", Reg. No. CZ.02.1.01/0.0/0.0/16_025/0007293.

CONFLICT OF INTERESTS

The authors would like to confirm that there is no conflict of interests associated with this publication and there is no financial fund for this work that can affect the research outcomes.

REFERENCES

[1] Minh V.T. (2012) Advanced Vehicle Dynamics, Publisher: Kuala Lumpur, Universiti of Malaya Press, Malaysia.

[2] Zhao P., Chen J., Song Y., Tao X., Xu T. and Mei T. Design of a Control System for an Autonomous Vehicle Based on Adaptive-PID, *International Journal of Advanced Robotic Systems*, 2012; 9(44); 1-11.

[3] Taghavifar H. and Rakheja S. Path-tracking of autonomous vehicles using a novel adaptive robust exponential-like-sliding-mode fuzzy type-2 neural network controller, *Mechanical Systems and Signal Processing*, 2019; 130; 41-55.

[4] Hu C., Wang Z., Taghavifar H., Na J., Qin Y., Guo J. and Wei C. MME-EKF-Based Path-Tracking Control of Autonomous Vehicles Considering Input Saturation, *IEEE Transactions on Vehicular Technology*, 2019; 68(6); 5246-5259.

[5] Wang Y., Gao S., Wang Y., Wang P, Zhou Y. and Xu Y. Robust Trajectory Tracking Control for Autonomous Vehicle Subject to Velocity-Varying and Uncertain Lateral Disturbance, *Archives of Transport*, 2021; 57(1); 7-23.

[6] Calzolari D., Schürmann B. and Althoff M. Comparison of trajectory tracking controllers for autonomous vehicles, *In*: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC),* Yokohama, Japan, 2017; p. 1-8.

[7] Varma B., Swamy N. and Mukherjee S. Trajectory Tracking of Autonomous Vehicles using Different Control Techniques (PID vs LQR vs MPC), *In: 2020 International Conference on Smart Technologies in Computing, Electrical and Electronics (ICSTCEE),* Bengaluru, India, 2020; p. 84-89.

[8] Jezierski A., Mozaryn J., Suski D., A Comparison of LQR and MPC Control Algorithms of an Inverted Pendulum. *In*: *2017 Trends in Advanced Intelligent Control, Optimization and Automation*, Kraków, Poland, 2017; p. 65-76.

[9] Alcala E., Puig V., Quevedo J., Escobet T. and Comasolivas C. Autonomous vehicle control using a kinematic Lyapunov-based technique with LQR-LMI tuning, *Control Engineering Practice*, 2018; 73; 1-12.

[10] Lee K., Jeon S., Kim H. and Kum D. Optimal Path Tracking Control of Autonomous Vehicle: Adaptive Full-State Linear Quadratic Gaussian (LQG) Control, *IEEE Access*, 2019; 7; 109120-109133.

[11] Vu T.M. and Nitin A., Robust Model Predictive Control for Input Saturated and Softened State Constraints, *Asian Journal of Control*, 2005; 7(3); 319-325.

[12] Minh V.T. Nonlinear Model Predictive Controller and Feasible Path Planning for Autonomous Robots, *Open Comput. Sci*. 2016; 6, pp. 178-186.

[13] Reda A., Ahmed Bouzid A. and József Vásárhelyi J. *Acta Polytechnica Hungarica*, 2020; 17(7); 163-182.

[14] Geng K. and Liu S. Robust Path Tracking Control for Autonomous Vehicle Based on a Novel Fault Tolerant Adaptive Model Predictive Control Algorithm, *Applied Sciences*, 2020; 10(18); 1-20.

[15] Chen S., Chen H., and Negrut D. Implementation of MPC-Based Trajectory Tracking Considering Different Fidelity Vehicle Models, *Journal of Beijing Institute of Technology*, 2020; 29(3); 303-316.

[16] Marcano M., Díaz S., Pérez J. and Irigoyen E. A Review of Shared Control for Automated Vehicles: Theory and Applications, *IEEE Transactions on Human-Machine Systems*, 2020; 50(6); 475-491.

[17] Chen S. and Chen H. MPC-based path tracking with PID speed control for autonomous vehicles, *J. Cent. South Univ.*, 2020; 27; 3702–3720.

[18] Minh V.T., Moezzi R., Dhoska K. and Pumwa J. Model Predictive Control for Autonomous Vehicle Tracking, *International Journal of Innovative Technology and Interdisciplinary Sciences*, 2021; 4(1); 560-603.

[19] Wang J., Teng F., Li J., Zang L., Fan T., Zhang J. and Wang X. Intelligent vehicle lane change trajectory control algorithm based on weight coefficient adaptive adjustment, *Advances in Mechanical Engineering*, 2021; 13(3); 1-16.

[20] Cao H. and Zoldy M. MPC Tracking Controller Parameters Impacts in Roundabouts, *Mathematics*, 2021; 9; 1-18.

[21] Minh V.T. and Hashim F. Tracking setpoint robust model predictive control for input saturated and softened state constraints, *International Journal of Control, Automation and Systems*, 2011, 9(5), 958-965.

[22] Minh V.T. and Pumwa J. Feasible Path Planning for Autonomous Vehicles, *Mathematical Problems in Engineering*, 2014; 2014; 1-12.

[23] Minh V.T., Mohd Hashim F.B. and Awang M. Development of a real-time clutch transition strategy for a parallel hybrid electric vehicle. *Proc. Inst. Mech. Eng. Part I: J. Syst. Control. Eng.* 2012; 226(2); 188-203.

[24] Minh V.T., Afzulpurkar N. and Wan Muhamad W.M. Fault detection model-based controller for process systems. *Asian J. Control*, 2011;13(3); 382-397.

[25] Minh V.T., Tamre M., Musalimov V., Kovalenko P., Rubinshtein I., Ovchinnikov I. Reza Moezzi. Simulation of Human Gait Movements. *International Journal of Innovative Technology and Interdisciplinary Sciences*, 2020; 3(1); 326-345.

[26] Minh V.T., Afzulpurkar N. and Muhamad W.M.W. Fault detection and control of process systems. *Math Probl. Eng.*, 2007; 2007; 1-20.

[27] Minh V.T. Automatic Control of Clutch Engagement and Slip for Hybrid Vehicle. *International Journal of Innovative Technology and Interdisciplinary Sciences*, 2019; 2(1); 49-61.

[28] Ovchinnikov I, Kovalenko P. Predictive Control Model to Simulate Humanoid Gait: Predictive Control Model to Simulate Humanoid Gait. *International Journal of Innovative Technology and Interdisciplinary Sciences*, 2019; 1(1); 9-17.

[29] Pumwa J. Time Variant Predictive Control of Autonomous Vehicles: Time Variant Predictive Control of Autonomous Vehicles. International Journal of Innovative Technology and Interdisciplinary Sciences, 2019; 2(1); 62-77.