Research Article Open Access

Vu Trieu Minh

Nonlinear Model Predictive Controller and Feasible Path Planning for Autonomous Robots

DOI 10.1515/comp-2016-0015
Received Oct 01, 2016; accepted Oct 24, 2016

Abstract: This paper develops the nonlinear model predictive control (NMPC) algorithm to control autonomous robots tracking feasible paths generated directly from the nonlinear dynamic equations. NMPC algorithm can secure the stability of this dynamic system by imposing additional conditions on the open loop NMPC regulator. The NMPC algorithm maintains a terminal constrained region to the origin and thus, guarantees the stability of the nonlinear system. Simulations show that the NMPC algorithm can minimize the path tracking errors and control the autonomous robots tracking exactly on the feasible paths subject to the system's physical constraints.

Keywords: Nonlinear model predictive control; autonomous robots; nonlinear dynamic system; feasible path generation; system stabilization

1 Introduction

Autonomous robots have been studied and developed extensively in recent years for new systems that can be able to drive the robot automatically from any start points to any destination points generated online from the global positioning system (GPS) maps and subject to the robot physical constraints and obstacles. This paper develops a nonlinear model predictive control (NMPC) scheme for controlling autonomous robot tracking referenced paths generated online and to control the robot to track on them. This system can be used for unmanned ground robots or for auto-parking/driving passenger robots.

Motivation for the use of model predictive control (MPC) controller is its ability to handle the constraints online within its open-loop optimizer control calculations while most of other control techniques are unfavorable in handling the online constraints and try to avoid them, thus, losing the best achievable performance. MPC can provide the real-time optimal inputs and make the close loop system operating near the constrained limits and hence, yield much better performances.

To deal with the system uncertainties and the model-plant mismatches, some robust MPC schemes are being developed accounting for the modeling errors at the controller design. Robust MPC can forecast all possible models in the plant uncertainty set and then, an optimal action can be determined through the min-max optimization. Schemes for robust MPC tracking setpoints can be read in Minh V and Hashim F (2011) [1], where the system uncertainties are represented by a set of multiple models via a tree trajectory and its branches. And the robust MPC problem is to find the optimal control actions that, once implemented, cause all branches to converge to a robust control invariant set.

To deal with the realistic movements of autonomous robots, some mathematical algorithms for finding feasible paths of nonholonomic robots including flatness, polynomial, and symmetric polynomial trajectories subject to the robot dynamical constraints and obstacles can be read in Minh V and Pumwa J (2014) [2].

Online generation of feasible paths for autonomous ground robots from the GPS maps and/or from their onboard 3D cameras have been studied in a number of articles. The robot can go off road subject to the robot physical constraints on speed, steering angle, and the environmental obstacles, etc. A recent genetic algorithm to optimize the path for navigation of agricultural mobile robots in fields involving obstacles can be read in [3].

A new path tracking algorithm using the future prediction control for autonomous robots can be read in [4] for nonlinear nonholonomic dynamic equations. This approach consists of steering wheel control and lateral movement. This control algorithm is applied for the nonholonomic navigation problem to track a referenced trajectory. The controller consists of the relationship among the future lateral error, the linear velocity, the heading error, and the referenced yaw rate.

NMPC algorithms are referred in Minh V and Afzulpurkar N (2006) [5], where three NMPC performances

Vu Trieu Minh: Department of Mechatronics Tallinn University of Technology, Tallinn, Estonia; Email: vutrieuminh@yahoo.com

with zero terminal region, quasi-infinite horizon, and softened state constraints are presented and compared. In these NMPC schemes, all the online inputs solution from the NMPC optimizer is implemented for the close-loop control by solving on-line the robot nonlinear dynamics ODEs.

There have been already several usefully control techniques to control the robot tracking desired paths. However the idea of using a full MPC controller for robot tracking online from the robot nonlinear dynamic equations (ODEs) with time variant system is still missing. A study of a simple MPC scheme for autonomous ground robot can be read in Falcon P. *et al.* (2008) [6], where an initial work based on MPC controller is presented. However, this paper failed to mention the real-time solving of the robot nonlinear dynamic equations (ODEs) and the calculation to obtain the optimal inputs for the robot velocity and its steering velocity. Similarly, another paper used MPC controller for autonomous robots tracking is presented by Lei L. *et al.* (2011) [7], where the robot movements are approximately linearized from the robot coordinates and the heading angle. The paper failed to include the steering angle limits and other physical constraints into its dynamic equations.

A paper on MPC for mobile robot trajectory control can be read in Baharonian M. *et al.* (2011) [8]. This paper comes out with an assumption that there is already a virtual referenced trajectory and then, the control problem becomes too simple and trivial. Another adaptive trajectory tracking control of wheeled mobile robot is developed by Wang J. *et al.* (2011) [9]. However, this paper did not mention how a feasible trajectory can be generated and how an optimal control action can be achieved for the best tracking performance. Another article by Shim T. *et al.* (2012) [10] derives a NMPC to control the front steering and the wheel torque for an autonomous ground robot. However, the paper failed to apply the on-line solving of the nonlinear dynamics ODEs for this NMPC.

Therefore, this paper develops a comprehensive NMPC scheme for tracking referenced trajectories generated online by nonlinear ODEs from the robot dynamics subject to its constraints and obstacles. The robot position data can be updated online from the GPS or the onboard 3D camera, and then, the feasible trajectory paths can be automatically generated onboard subject to the robot constraints on speed, steering angle, sideslip, and the environmental obstacles, etc.. This paper is organized as follows: Section 2 describes the system modeling; Section 3 develops NMPC algorithms; Section 4 presents tracking simulations; And finally, some study remarks are concluded in section 5.

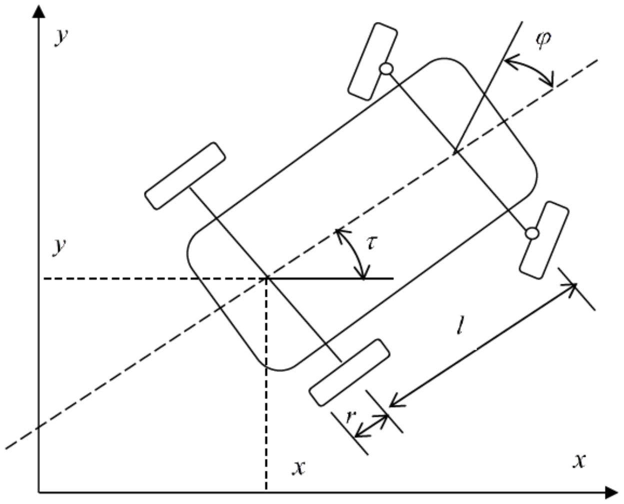

Figure 1: A simplified robot model.

2 System modeling

Consider a robot rolling without slipping, the robot dynamics can be written in a set of first-order differential equations from its configuration variables. If the robot has the rear-wheel driving, the robot kinematic model, shown in figure 1, can be derived in equation (1):

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\tau}\\ \dot{\varphi}\end{bmatrix} = \begin{bmatrix}\cos\tau\\ \sin\tau\\ \frac{\tan\varphi}{l}\\ 0\end{bmatrix} u_1 + \begin{bmatrix}0\\0\\0\\1\end{bmatrix} u_2 \tag{1}$$

In figure 1, r is the robot wheel radius and l is the base length; x and y are the Cartesian coordinates of the rear wheel, τ measures the orientation of the robot body with respect to the x axis, and φ is the steering angle. The robot motion is controlled by two inputs, u_1 is the linear driving velocity $\upsilon = \theta r$ (θ is the vehicle wheel angular velocity), and, u_2 is the angular steering velocity (φ). There are four (4) state variables, namely the position of the robot $x_1 = x$ and $x_2 = y$; the angle of the robot body orientation with respect to the x axis, $x_3 = \tau$; and the steering angle, $x_4 = \varphi$.

The system in (1) is a nonholonomic and if $\varphi \neq \frac{\pi}{2}$, this system is fully controllable. This robot model is nonlinear and has the first order derivative form:

$$\dot{X} = f(x, u) \tag{2}$$

where the state variables are $x = [x, y, \tau, \varphi]'$, and the inputs are $u = [u_1, u_2]'$. The nonlinear equation in (2) can be expanded in Taylor series around the reference setpoints

(x_r, u_r) at $\dot{X}_r = f(x_r, u_r)$, that:

$$\dot{X} \approx f(x_r, u_r) + f_{x,r}(x - x_r) + f_{u,r}(u - u_r) \tag{3}$$

where $f_{x.r}$ and $f_{r.x}$ are the Jacobean of f corresponding to x and u, evaluated around the reference setpoints (x_r, u_r).

Subtraction of (3) and $\dot{X}_r = f(x_r, u_r)$ results a linearized system in continuous time (t):

$$\dot{\tilde{X}}(t) = A(t)\tilde{X}(t) + B(t)\tilde{u}(t) \tag{4}$$

where $\tilde{X}(t) = X(t) - X_r(t)$, and $\tilde{u}(t) = u(t) - u_r(t)$,

$$A(t) = \begin{bmatrix} 0 & 0 & -u_{r1}(t)\sin\tau_r(t) & 0 \\ 0 & 0 & u_{r1}(t)\cos\tau_r(t) & 0 \\ 0 & 0 & 0 & \frac{u_{r1}(t)}{l\cos^2\varphi_r(t)} \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$$B(t) = \begin{bmatrix} \cos\tau_r(t) & 0 \\ \sin\tau_r(t) & 0 \\ \frac{\tan\varphi_r(t)}{l} & 0 \\ 0 & 1 \end{bmatrix},$$

The continuous time system in (4) can be transformed to a discrete-time (k) with a sampling interval, $k + 1 = k + \Delta t$, and, Δt is the length of the sampling interval. The inputs $u(k)$are held constant during this time interval $(k + 1)$ and (k). The symbols of $x_k = x(k)$ and $u_k = u(k)$ are also used:

$$\tilde{X}(k+1) = A(k)\tilde{X}(k) + B(k)\tilde{u}(k)\tilde{Y}(k) \quad = C(k)\tilde{X}(k) \tag{5}$$

where,

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

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

$$C(k) = [1]\,, \text{ and, } \tilde{X}(k) = X(k) - X_r(k),\ \tilde{u}(k) = u(k) - u_r(k).$$

In this discretized form, the two control inputs are the difference in the actual and the desired velocity, $u_1(k) - u_{r1}(k)$, and, $u_2(k) - u_{r2}(k)$. The four outputs, $y(k) = \tilde{Y}(k) = C(k)\tilde{X}(k)$, are assumed to be measured and updated in real-time from the GPS. It is noted that the robot discretized model in (5) is a time variant system and its transfer function is depending on the dynamic positions and the sampling speeds. Linearized equations (5) are used to develop MPC schemes in the next part.

3 Model Predictive Control Design

This part presents the design of NMPC algorithms for the discretized linearized model. The NMPC works out based on the fundamental MPC objective functions for hard constraints, softened constraints and constraints in regions.

For the MPC with hard constraints, from (5), the prediction horizon for the outputs, $y_{k+i|k}$, and the input increments, $\Delta u_{k+i|k}$, can be rewritten as,

$$\begin{bmatrix} y_{k+1|k} \\ y_{k+2|k} \\ \vdots \\ y_{k+N_y|k} \end{bmatrix} = \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^{N_y} \end{bmatrix} x_{k|k} + \begin{bmatrix} CB \\ CAB + CB \\ \vdots \\ \sum_{i=1}^{N_y} CA^{i-1}B \end{bmatrix} u_{k-1}$$

$$+ \begin{bmatrix} CB & 0 & \cdots & 0 \\ CAB + CB & CB & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^{N_y} CA^{i-1}B & \sum_{i=1}^{N_y-1} CA^{i-1}B & \cdots & \sum_{i=1}^{N_y-N_u+1} CA^{i-1}B \end{bmatrix}$$

$$\cdot \begin{bmatrix} \Delta u_k \\ \Delta u_{k+1} \\ \vdots \\ \Delta u_{k+N_u-1} \end{bmatrix}$$

then, the tracking setpoints MPC objective function with hard constraints is:

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

subject to:

$$\begin{aligned} & u_k \in \mathcal{U}, \text{ and } u_{k+i} \in [u_{\min}, u_{\max}]\,, \\ & \Delta u_{k+i} \in [\Delta u_{\min}, \Delta u_{\max}]\,, \text{ for } i = 0, 1, \ldots, N_u - 1, \\ & y_k \in \mathcal{Y}, \text{ and } y_{k+i|k} \in [y_{\min}, y_{\max}]\,, \\ & \text{for } i = 0, 1, \ldots, N_y - 1, \\ & \Delta u_k = u_k - u_{k-1} \in \Delta\mathcal{U}, \text{ and } \Delta u_{k+i} = 0, \text{ for } i \geq N_u, \\ & x_{k|k} = x(k),\ x_{k+i+1|k} = A(k)x_{k+i|k} + B(k)u_{k+i}, \\ & u_{k+i|k} = u_{k+i-1|k} + \Delta u_{k+i|k},\ y_{k+i|k} = C(k)x_{k+i|k}. \end{aligned} \tag{6}$$

where $x(k)$ denotes the state variables at the current discrete time (k) : $U \overset{\Delta}{=} \{\Delta u_k, \ldots, \Delta u_{k+N_{u-1}}\}$ is the solution of input increments, N_u is the inputs predictive horizon; N_y is the outputs predictive horizon; $y_{k+i|k}$ are the predictive outputs at the current discrete time (k), $r_{k+i|k}$ are the corresponding reference output setpoints; $\Delta u_{k+i|k}$ are the input increments prediction with $\Delta u_{k+i|k} = u_{k+i|k} - u_{k+i-1|k}$;

$Q = Q' \geq 0$, $R = R' > 0$ are the weighting penalty matrices for predicted outputs and input increments, respectively.

By substituting $x_{k+N_y|k} = A^{N_y}(k)x(k) + \sum_{i=0}^{N_y-1} A^i(k)B(k) u_{k+N_y-1-i}$, equation (6) can be rewritten as a function of only the current state $x(k)$ and the current setpoints $r(k)$:

$$\Psi\left(x(k), r(k)\right) = \frac{1}{2}x'(k)Yx(k) \qquad (7)$$
$$+ \min_{U}\left\{\frac{1}{2}U'HU + x'(k)r(k)FU\right\}$$

subject to the hard combined constraints of $GU \leq W + Ex(k)$, where the column vector $U \overset{\Delta}{=} \left[\Delta u_k, \ldots, \Delta u_{k+N_p-1}\right]' \in \Delta\mathcal{U}$ is the prediction optimization vector; $H = H' > 0$, and H, F, YG, W and E are matrices obtained from Q, R and given constraints in (6). As only the optimizer U is needed, the term involving Y is usually removed from (7).

For the MPC with softened constraints, if the outputs constraints are the tracking positions which are not strictly imposed and can be violated somewhat during the evolution of the performance. To guarantee the system stability once the outputs violate the constraints, the hard constrained optimization in (6) can be modified to a new MPC objective function with softened constraints as:

$$\min_{U \overset{\Delta}{=} \{\Delta u_k, \ldots, \Delta u_{k+N_u-1}\}} \left\{ J(U, x(k)) = \sum_{i=0}^{N_y-1}\left[(y_{k+i|k} - r_{k+i|k})' \right.\right. \qquad (8)$$
$$\left.\left. \cdot Q(y_{k+i|k} - r_{k+i|k}) + \Delta u'_{k+i|k}R\Delta u_{k+i|k} + \varepsilon'_i(k)\Lambda\varepsilon_i(k)\right]\right\}$$

where $\varepsilon_i(k) \geq 0$ are the new penalty terms added to the MPC objective function, $\varepsilon_i(k) = [\varepsilon_y; \varepsilon_u]$, $y_{\min} - \varepsilon_y \leq y_{k+i|k} \leq y_{\max} + \varepsilon_y$ and $u_{\min} - \varepsilon_u \leq u_{k+i|k} \leq u_{\max} + \varepsilon_u$. And $\Lambda = \Lambda' \geq 0$ is the new penalty matrix (usually $\Lambda > 0$ and set with small values). These terms, $\varepsilon_i(k)$, will keep the constrained violations at low values until the solution is returned. A new MPC algorithm for softened constraints to select the optimal inputs $u^\star(k + i|k)$ can be conducted similarly to (7) with the new added penalty terms $\varepsilon'_i(k)\Lambda\varepsilon_i(k)$.

For the MPC with constraints in regions, robustness of MPC can be also increased if some setpoints can be relaxed into regions rather than in some specific values. Then, a new MPC algorithm can be developed if the setpoints $r(k)$ now can be changed into some regions. An output region is defined by the minimum and maximum values of a desired range. The minimum value is the lower limit, and the maximum value is the upper limit and satisfied $y_{lower} \leq y_{k+i|k} \leq y_{upper}$. The modified objective function

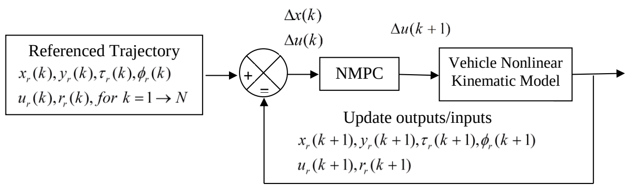
$\Delta x(k)$
$\Delta u(k)$
$\Delta u(k+1)$
Referenced Trajectory
$x_r(k), y_r(k), \tau_r(k), \phi_r(k)$
$u_r(k), r_r(k), for\ k = 1 \rightarrow N$
+
=
NMPC
Vehicle Nonlinear Kinematic Model
Update outputs/inputs
$x_r(k+1), y_r(k+1), \tau_r(k+1), \phi_r(k+1)$
$u_r(k+1), r_r(k+1)$

Figure 2: NMPC control system.

for the MPC with output regions is:

$$\min_{U \overset{\Delta}{=} \{\Delta u_k, \ldots, \Delta u_{k+N_u-1}\}} \left\{ J(U, x(k)) = \sum_{i=0}^{N_y-1}\left[z'_{k+i|k}Qz_{k+i|k} \right.\right. \qquad (9)$$
$$\left.\left. + \Delta u'_{k+i|k}R\Delta u_{k+i|k}\right]\right\}$$

where $z_{k+i|k} \geq 0$; $z_{k+i|k} = y_{k+i|k} - y_{upper}$ for $y_{k+i|k} > y_{upper}$; $z_{k+i|k} = y_{lower} - y_{k+i|k}$ for $y_{k+i|k} < y_{lower}$; $z_{k+i|k} = 0$ for $y_{lower} \leq y_{k+i|k} \leq y_{upper}$

As long as the outputs still lie inside the desired regions, no control actions are taken because none of the control objectives have been violated, all $z_{k+i|k} = 0$. But when an output violates the desired region, the control objective in the MPC regulator will activate and push them back to the desired regions.

As per Minh V.T and Afzulpurkar N (2006) [5], the NMPC schemes now can be developed from MPC schemes in (6), (8), and (9) by imposing some additional stabilized conditions on the open-loop optimal regulator. These conditions make a terminal constrained region to the origin. For the NMPC perfromances in this paper, we apply the zero terminal equality or zero terminal region at the end of the prediction horizon *i.e.* adding the zero constraint for the terminal prediction state at $x_{k+i|k} = A^i(k)x_{k|k} + \sum_{i \geq N_u}^{N_y-1} A^i(k)B(k)u_{k+N_y-1-i|k} = 0$ in the MPC objective functions. Numerical experiments of the NMPC performances are presented in the next part.

4 NMPC performances

The diagram of the NMPC for the robot nonlinear kinematic model is shown in figure 2.

The online optimization problem for this NMPC is taken place at real-time (within some milliseconds depending on the complexity of the system and the power of the computer). The NMPC regulator determines an optimal future input trajectory and then implements only the first element of the current inputs for the close-loop control by

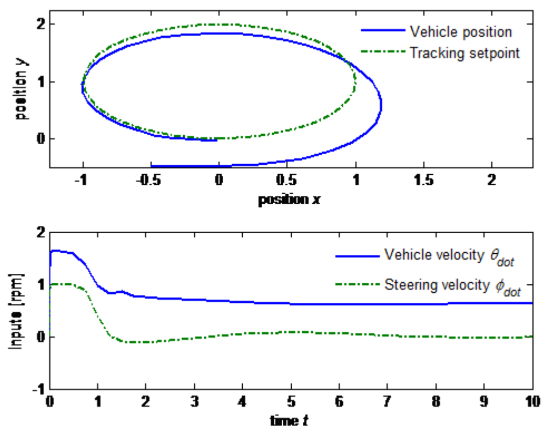

Figure 3: NMPC tracking for a full circle.

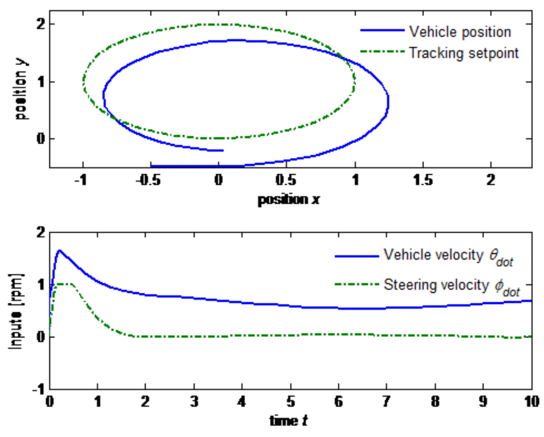

Figure 4: NMPC with shortened horizon.

solving directly on-line the ordinary differential equations (ODEs) for the robot kinematics in (2) repeatedly

NMPC for a full circle trajectory

We now use the NMPC regulator to run the autonomous robot to track a full circular trajectory. For this simulation, the initial positions of the robot are set at $X_0 = \begin{bmatrix}-0.5 & -0.5 & 0 & 0\end{bmatrix}'$; The constraints are set at $u_{\min} = [-1, -1]'$, $u_{\max} = [1, 1]'$, $\Delta u_{\min} = [-0.5, -0.5]'$, $\Delta u_{\max} = [0.5, 0.5]'$, $y_{\min} = [-1, -1, -1, -1]'$, and $y_{\max} = [1, 1, 1, 1]'$; The predictive horizons are set at $N_u = 10$ and $N_y = 10$; Penalty matrices are set at $Q = diag\{1, 1, 1, 1\}$ and $R = diag\{1, 1\}$. Performance of the NMPC is shown in figure 3. The NMPC optimizer minimizes the tracking errors and tracks the robot with very small errors left at the end of the trajectory. The inputs look also good since they are physically smooth enough for controlling the robot.

If the prediction horizon is shortened, the calculation burden will be considerably reduced but will lead to faster changes in the inputs, then, cause bad performance of the outputs. With short prediction horizons, the system may become instable. Figure 4 shows the NMPC performance with shortened horizons to $N_u = 4$ and $N_y = 4$. We can see the worse performance from the final tracking errors and the sharp inputs movement at the start time.

In NMPC, we can regulate the control performance by changing the predictive horizon length, penalty matrices, softened constraints or time scanning intervals. We can also regulate by changing reference setpoint errors ($y_{k+i|k} - r_{k+i|k}$) to offset the robot sideslip or to compensate the model-plant mismatches. To offset the robot sideslips or the model-plant mismatches, we can dynamically change

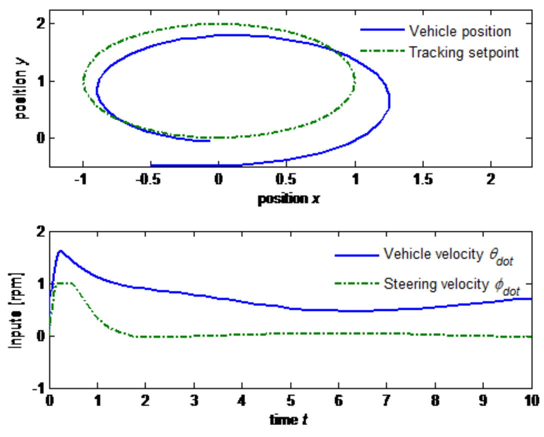

Figure 5: NMPC with new setpoint offsets.

these setpoint errors. For example, if we set the setpoint errors at $r_k = [-0.1, 0.3, 0, 0]'$, the NMPC performance is shown in figure 5 and we can see some better tracking performances:

In the next part, we will investigate the ability of the NMPC to track other feasible trajectories generated directly from the robot kinematic differential equations.

NMPC for tracking flatness trajectory

Generation of flatness trajectory equations is presented [2]. Figure 6 shows a flatness trajectory for the robot from the initial position, $[x_0, y_0] = [0, 0]$, to the final position, $[x_F, y_F] = [10, 10]$, and the development of the orientation angle, θ, and steering angle, φ, during the travel. The time for completing this travel is set for $T = 100$ sec;

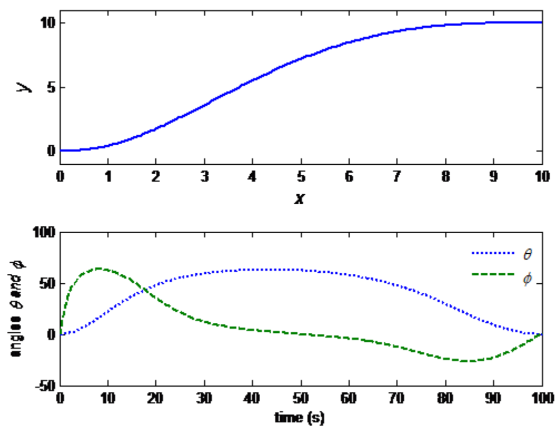

Figure 6: Flatness trajectory reference setpoints.

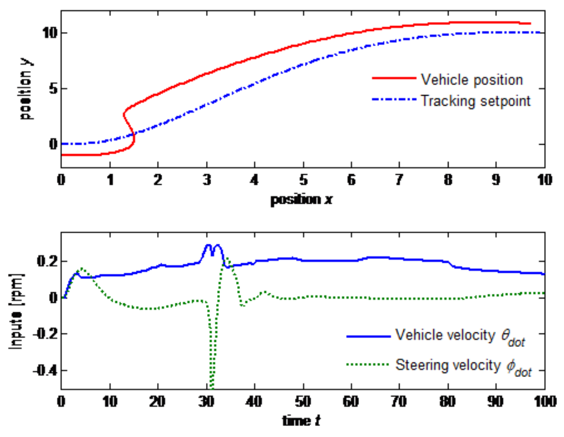

Figure 8: NMPC with too short horizon.

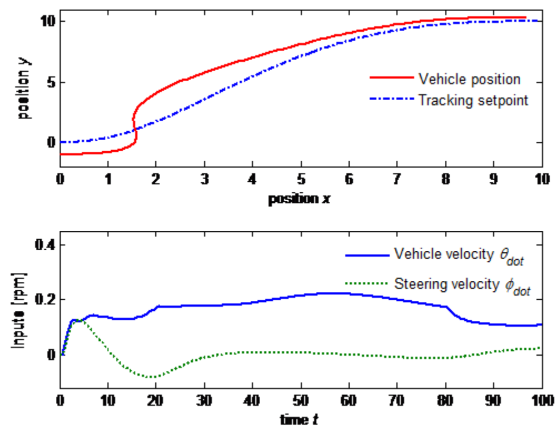

Figure 7: NMPC for tracking flatness trajectory.

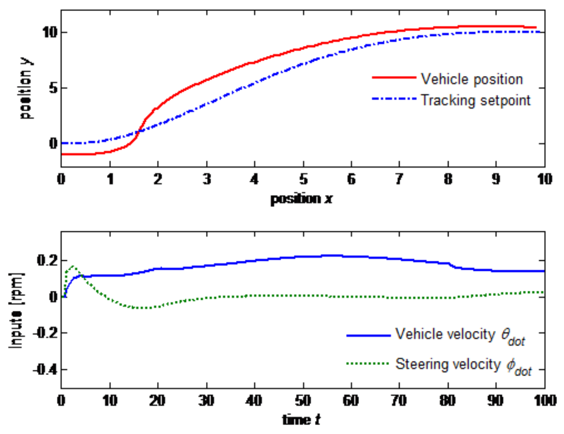

Figure 9: NMPC with $R = diag\{1, 1\}$.

For this NMPC tracking, the initial positions of the robot are set at $X_0 = \begin{bmatrix} 0 & -0.5 & 0 & 0 \end{bmatrix}'$; The predictive horizons are set at $N_u = 10$ and $N_y = 10$; Penalty matrices are set at $Q = diag\{1, 1, 1, 1\}$ and $R = diag\{60, 60\}$; The reference velocity inputs are set at, $u_1 = \frac{1}{3}$ (u_1 is set at 0 at starting point, during the first 1/5 time length, u_1 will gradually increase and maintain at 1/3 for 60 sec, during the final 1/5 time length, u_1 will decrease back to 0), and, $u_2 = 0$. Performance of this NMPC tracking is shown in figure 7. The robot starts with an initial velocity of $u_1 = 0$ and from the initial position of $X_0 = \begin{bmatrix} 0 & -0.5 & 0 & 0 \end{bmatrix}'$ gradually tracks to the reference tracking trajectory in 15 sec.

Next, we shorten the horizon prediction length to $N_u = 6$ and $N_y = 6$ while maintain other parameters unchanged. We can see that this shortened prediction horizon can degrade the performance because it causes the deterioration of the inputs. In this example, the system becomes unstable (infeasible) for the inputs as shown in figure 8.

The above instable system is due to the too heavy penalty values imposed on the input matrix ($R = diag\{60, 60\}$). This heavy penalty causes too slow and too small changes in the inputs. If we release this penalty on the inputs to $R = diag\{1, 1\}$, the system returns stable as shown in figure 9.

The above NMPC can run with longer predictive horizon and achieve better performance. Figure 10 shows this NMPC performance with $N_u = 16$ and $N_y = 16$. We can see the very small tracking errors at the end of the travel.

Next, we continue to test the NMPC for tracking the polynomial trajectory since the polynomial trajectories can be generated faster and smoother than the flatness trajectories.

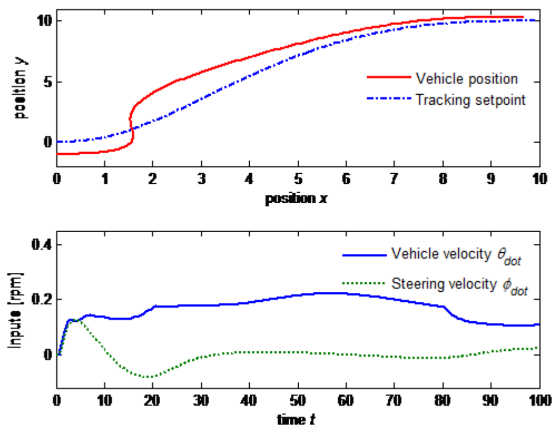

Figure 10: NMPC with $N_u = 16$ and $N_y = 16$.

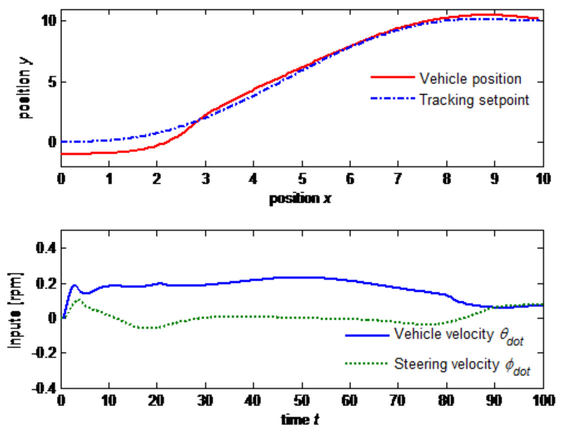

Figure 12: NMPC for tracking polynomial trajectory.

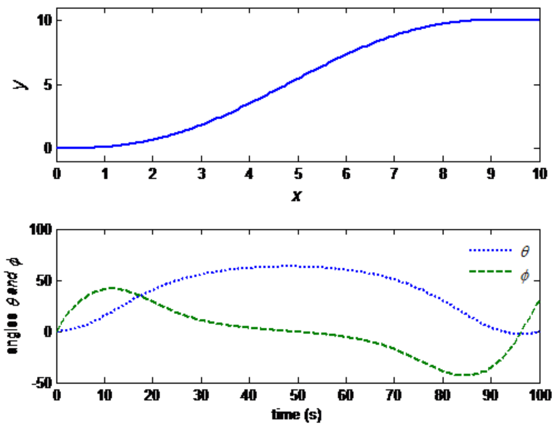

Figure 11: Polynomial trajectory references.

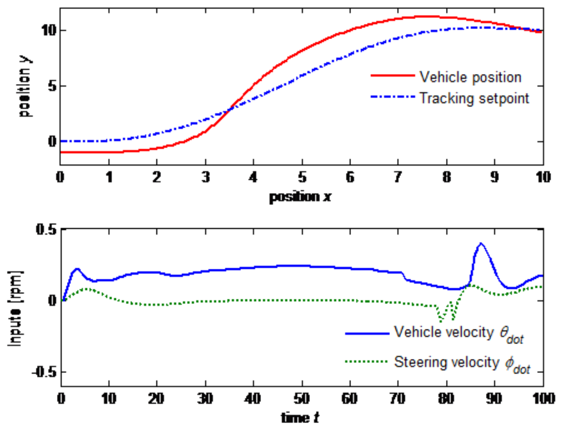

Figure 13: NMPC with shorten horizon

NMPC for tracking polynomial trajectory

Generation of polynomial trajectory is presented in [2]. Figure 11 shows a polynomial trajectory for the robot from the initial position, $[x_0, y_0] = [0, 0]$, to the final position, $[x_F, y_F] = [10, 10]$, and the development of its orientation angle, θ, and steering angle, φ, during the travel. Similarly, the time for completing this travel is set for $T = 100$ sec;

The initial positions of the robot are set at $X_0 = \begin{bmatrix} 0 & -0.5 & 0 & 0 \end{bmatrix}'$; The predictive horizons are set at $N_u = 10$ and $N_y = 10$; Penalty matrices are set at $Q = diag\{1, 1, 1, 1\}$ and $R = diag\{60, 60\}$; The reference velocity inputs are set at $u_1 = 1$ (u_1 is set at 0 at the starting point, during the first 1/5 time length, u_1 will gradually increase and maintain at 1, during the final 1/5 time length, u_1 will decrease back to 0), and, $u_2 = 0$. Performance of this NMPC tracking is shown in figure 12. The robot gradually tracks exactly the reference setpoints in 28 sec.

Next, we shorten the horizon prediction length to $N_u = 5$ and $N_y = 5$, we can see that the too short prediction horizon can degrade the performance. The system becomes instable as shown in figure 13. The performance is worse due to the sensitiveness of the inputs and the robot cannot reach the output setpoints.

The above shortened horizon NMPC becomes instable at the end of the trajectory since the input increments are too slow and too small due to the too heavy penalty imposed on the inputs matrix, $R = diag\{60, 60\}$. If we release this penalty and the inputs can variate more freely, the system will return stable with $R = diag\{1, 1\}$ as shown in figure 14.

Now we can lengthen the predictive horizon for the above NMPC to $N_u = 16$ and $N_y = 16$. The system performance becomes much better as shown in figure 15. The

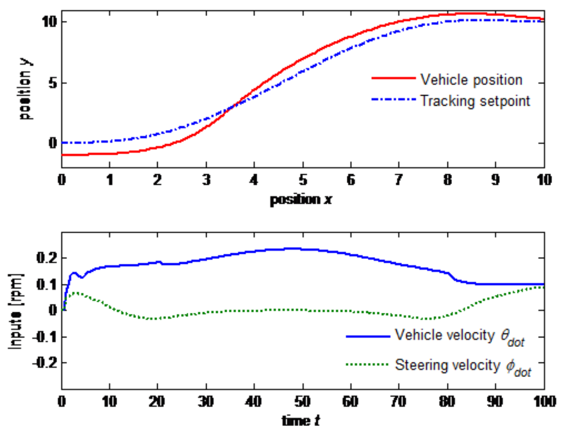

Figure 14: NMPC with $R = diag\{1, 1\}$.

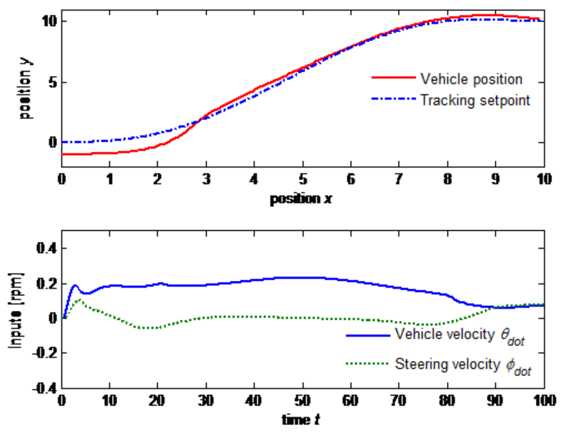

Figure 15: NMPC with lengthen horizon.

robot tracks rapidly and exactly on the reference setpoints in 28 sec.

The above NMPC performances show that the robot can track all trajectories from the different initial positions. This difference can be considered as the possible errors of the measured outputs or the initial model-plant mismatches. NMPC regulator can overcome those errors and track the robot exactly along the desired reference setpoints.

5 Conclusion

In this paper, NMPC schemes for tracking setpoints have been developed and tested for controlling the robot tracking to different trajectories. Simulations show that NMPC can control very well the tracking setpoints subject to constraints. The NMPC performance, stabilization as well as the robustness can be regulated and improved by varying NMPC parameters as well as modifying NMPC objective functions to softened constraints or to output regions. NMPC schemes are able to guarantee the system stability even when the initial conditions lead to violations of some constraints.

Even though simulations show that NMPC algorithms are successful in controlling the robot tracking, model of uncertainty and the model-plant mismatches that may affect to the closed loop stability are still open issues. Further analysis is needed for the effectiveness of the modified NMPC schemes to softened constraints and to output regions. Real experiments and other validations for this technique are also needed in the next step of the project.

Acknowledgement: The author would like to thank Tallinn University of Technology (TTU) for supporting this research project.

Conflict of Interests: The author declares that this article is our owned research property and there is no conflict of interests regarding the publication of this paper.

References

[1] Minh V.T., Hashim F.B., 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

[2] Minh V.T., Awang M., Parman S., Conditions for stabilizability of linear switched systems, International Journal of Control Automation and Systems, 2011, 9(1), 139–144

[3] Hameed I., Bochtis D., Sorensen C., An optimized field coverage planning approach for navigation of agricultural robots in fields involving obstacle areas, International Journal of Advanced Robotic Systems, 2013, Open Access, DOI: 10.5772/56248

[4] Zakaria M., Zamzuri H., Mamat R., Mazlan S., A path tracking algorithm using future prediction control with spike detection for an autonomous robot, International Journal of Advanced Robotic Systems 2013, Open Access, DOI: 10.5772/56658

[5] Minh V.T, Afzulpurkar N., Robust model predictive control for input saturated and softened state constraints, Asian Journal of Control, 2005, 7(3), 319–325

[6] Falcone P., Borrelli F., Tseng H., Asgari J., Hrovat D., A hierarchical model predictive control framework for autonomous ground robots, In: American Control Conference (ACC), Seattle, USA, June 2008, 3719–3724

[7] Lei L., Zhurong J., Tingting C., Xinchung J., Optimal model predictive control for path tracking of autonomous robot, In: 3rd International Conference on Measuring Technology and Mechatronics Automation (ICMTMA), Shanghai, China, February 2011, 2, 791–794

[8] Bahadorian M., Savkovic B., Eston R., Hesketh T., Toward a robust model predictive controller applied to mobile robot trajectory tracking control, In: Proceedings of the 18th IFAC World Congress, Milano, Italia, September 2011, 18(1), 552–557

[9] Wang J., Lu Z., Chen W., Wu X., An adaptive trajectory tracking control of wheeled mobile robots, In: Industrial Electronics and Applications (*ICIEA*), Beijing, China, June 2011, 1156–1160

[10] Shim T., Adireddy G., Yuan H., Autonomous robot collision avoidance system using path planning and model predictive control base active front steering and wheel torque control, Part D: Journal of Automobile Engineering, 2012, 226(6), 767–778