Stabilizing NMPC of Wheeled Mobile Robots Using Open-Source Real-Time Software

Mohamed W. Mehrez, George K. I. Mann and Raymond G. Gosine
Faculty of Engineering and Applied Science
Memorial University of Newfoundland
St. John's, NL, Canada
e-mail: {m.mehrez.said, gmann, rgosine}@mun.ca

***Abstract*— In this paper, a recently developed open-source toolkit implementing fast nonlinear model predictive control (NMPC) routines has been utilized to achieve the two main control objectives of nonholonomic mobile robots, namely, point stabilization and trajectory tracking. The stability of the controller has been guaranteed by adding final state equality constraint, which generally requires, for feasibility, long optimization horizon and hence is computationally demanding. In order to use the toolkit for real experiments, a C++ code, which couples the toolkit and a wheeled mobile robot research platform's software, has been developed. Full scale experiments have been conducted showing the applicability of the stabilizing terminal equality constraint NMPC, to wheeled mobile robots' control problems, in a real-time framework.**

Keywords—nonlinear model predictive control; stability; real-time nmpc; mobile robots; point stabilization; trajectory tracking

I. INTRODUCTION

Control problems related to nonholonomic mobile robots have gained a significant attention of researchers, over the past three decades. The reason for this interest arises from practical and theoretical concerns; the nonholonomic nature of this class of mobile robot imposes limitations on acceptable system velocities [1]. Nevertheless, the nonholonomy becomes usually useful as it limits the number of control inputs, while maintaining the full controllability of the system in the state space [2]. This advantage, however, imposes a difficulty that is connected with the point-stabilization task: the task cannot be accomplished with a pure feedback control algorithm [1].

The main two control objectives considered in the literature, of this class of mobile robots, are point-stabilization and trajectory tracking control. Point-stabilization or regulation intends to drive the robot between two poses: an initial pose and a target pose. On the other hand, trajectory tracking control aims at controlling robots to track a given time varying trajectory. Since it is relatively simpler, trajectory tracking problem has been extensively studied in the robotics society. In [3], several tracking techniques have been reported including dynamic feedback linearization, sliding mode, backstepping, and discontinuous controllers' techniques. Different studies considering point stabilization problem are also reported in [4], which include Lyapunov control, nonlinear geometric control, piecewise-continuous feedback, smooth time-varying control, and dynamic feedback linearization. Techniques achieving both control objectives include differential kinematic controller [5], feedback linearization [6], back stepping technique [7], and vector field orientation feedback control method [2].

Model predictive control (MPC) is one of the most frequently used advanced control techniques in the industry; the objective of MPC is to compute a future control sequence in a specified time horizon so that the prediction of the controlled system output is driven close to a reference value. This is achieved by minimizing a cost function with respect to future control actions through an online optimization step, where a set of control actions and system states constraints are satisfied [8]. MPC variants including linear MPC and nonlinear MPC (NMPC) have been used for solving the two main control objectives of mobile robots. Linear MPC uses a linearized version of the robot motion equations allowing it to be solely used for trajectory tracking problem [9], [10]; or its time-nonparameterized special case known as path following tracking [8], [11]. NMPC, which uses the explicit nonlinear motion model of robots, has been used for tracking problems [3], [12]; regulation problems [4], [13]; and both [14].

Due to the use of a predictive control horizon, the control stability becomes one of the main problems [4]. For a finite receding horizon, it was proven that the stability can be guaranteed by using terminal state equality constraints [15], [16]. Further work shows that the terminal-state equality constraint can be relaxed as a terminal-state inequality, by adding a terminal-state penalty [4], [3]. Another stability criteria based on first-state contractive predictive control has been presented in [14]. It was claimed in [4], [3], and [15] that achieving stability using the terminal equality constraint is time consuming and a practically intractable problem. Nonetheless, due to the advancements in computing machines and development of efficient numerical algorithms, a considerable number of dynamic optimization and NMPC implementation packages has been developed [17], [18]. A recently developed package (ACADO-Toolkit) [19], which implements real-time NMPC routines, has been proven to be open source, user

This research is supported by the Natural Sciences and Engineering Research Council of Canada (NESRC), the Research and Development Corporation (RDC), C-CORE J.I. Clark Chair, and Memorial University of Newfoundland.

friendly, extensible, self-contained, and computationally versatile, when compared with the previously developed optimization packages [19]. As per the previously outlined literature, it has been noticed that a study utilizing real-time NMPC for the two main control objectives of nonholomomic mobile robots, where stabilizing terminal equality constraints are considered, is needed.

In this work, ACADO-toolkit has been utilized to perform the two main control objectives of nonholonomic mobile robots, namely, point stabilization and trajectory tracking in an experimental context. Due to the computation versatility of this package, the computationally demanding stabilizing terminal equality constraints have been applied. A C++ code, which couples the used toolkit and Pioneer-3AT robots' input files, has been developed. Real-time experiments have been conducted showing the applicability of the final equality constraints in a real-time framework.

II. Preliminaries

This paper deals basically with the control problem of differential drive robots class of nonholonomic mobile robots. In this section, a brief description of this class of robots kinematics is presented together with an illustration of the control objectives: point stabilization and trajectory tracking.

A. *Robot Kinematics*

A differential drive mobile robot is a typical nonholonomic mobile robot; this class of robots can have the two wheels configuration, where a castor wheel is needed, or the four wheels configuration. The linear speeds of the right (v_r) and the left (v_l) wheels define the control inputs: linear speed ($v = (v_r + v_l)/ 2$) and angular speed ($\omega = (v_r + v_l)/ b$), where b is the wheel base. The kinematic equations of the robot under the nonholonomic constraint of rolling without slipping are given as follows [3]:

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

The state and control signal vectors are denoted as $\mathbf{x} = (x, y, \theta)^T$ and $\mathbf{u} = (v, \omega)^T$. As shown in Fig. 1(a), (x, y) represents the Cartesian position of the robot in the world frame of reference (X_I, Y_I), and (θ) represents the robot orientation measured from the positive (X_I) to the robot frame x-axis (x_R). Despite the simplicity of the model (1), it is sufficient to describe the nonholonomic constraints of the differential drive robots.

As system (1) has a driftless form (2) and the accessibility rank condition is globally satisfied [20], system given by (1) is controllable.

$$\dot{\mathbf{x}} = \begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{bmatrix} = \begin{bmatrix}\cos\theta\\ \sin\theta\\ 0\end{bmatrix} v + \begin{bmatrix}0\\ 0\\ 1\end{bmatrix}\omega \tag{2}$$

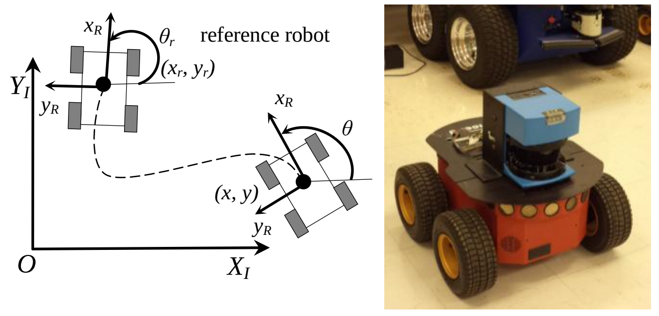

Fig. 1. (a) Differential drive robot kinematic. (b) Pioneer 3-At research platform

B. *Point stabilization and trajectory tracking*

In order to illustrate the two control problems of the mobile robot, a reference robot, shown in Fig. 1(a), is defined with a reference state vector $\mathbf{x}_r = (x_r, y_r, \theta_r)^T$ and a reference control vector $\mathbf{u}_r = (v_r, \omega_r)^T$, and subject to the same constraint as system (1). Thus, its kinematic motion model can be written as:

$$\dot{\mathbf{x}}_r = \begin{bmatrix}\dot{x}_r\\ \dot{y}_r\\ \dot{\theta}_r\end{bmatrix} = \begin{bmatrix} v_r\cos\theta_r \\ v_r\sin\theta_r \\ \omega_r\end{bmatrix} = \begin{bmatrix}\cos\theta_r & 0\\ \sin\theta_r & 0\\ 0 & 1\end{bmatrix}\mathbf{u}_r \tag{3}$$

At this point, the point stabilization problem is the case when the reference state vector $\mathbf{x}_r$ has a constant value corresponding to the desired goal position, and the control vector $\mathbf{u}_r$ has zero values for both linear and angular velocities reference. On the other hand, for the trajectory tracking problem, vectors $\mathbf{x}_r$ and $\mathbf{u}_r$ have time varying values depending on the chosen reference trajectory. In both cases, the control objective is to control (1) to track (3); thus, an error state vector $\mathbf{x}_e$ can be defined as:

$$\mathbf{x}_e = \begin{bmatrix}x_e\\ y_e\\ \theta_e\end{bmatrix} = \begin{bmatrix}\cos\theta & \sin\theta & 0\\ -\sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}x_r - x\\ y_r - y\\ \theta_r - \theta\end{bmatrix} \tag{4}$$

It can be easily seen that the two control objectives can be achieved by driving the state vector $\mathbf{x}_e$ to zero. The error dynamic model is obtained by differentiating (4) as:

$$\begin{aligned}\dot{x}_e &= \omega y_e - v + v_r\cos\theta_e\\ \dot{y}_e &= -\omega x_e + v_r\sin\theta_e\\ \dot{\theta}_e &= \omega_r - \omega\end{aligned} \tag{5}$$

A linearized version of the error model (5) has the following form:

$$\dot{\mathbf{x}}_e = \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} = \begin{bmatrix} 0 & \omega_r & 0 \\ -\omega_r & 0 & v_r \\ 0 & 0 & 0 \end{bmatrix} \mathbf{x}_e + \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \mathbf{u}_e \tag{6}$$

where $\mathbf{u}_e$ is defined as

$$\mathbf{u}_e = \begin{bmatrix} -v + v_r \cos\theta_e \\ \omega_r - \omega \end{bmatrix} \tag{7}$$

It can be easily checked that the controllability of model (6) is lost when the reference linear velocity and angular velocity values approach the origin, eliminating the applicability of point stabilization control.

At this point, it has to be mentioned that for point stabilization control purpose a motion model variant for (5) defined in the polar coordinates has been also considered in [4], and [14]; however, in the presented work here, model (5) has been solely used to achieve the two control objectives. Thus, the structure of the controller doesn't need to be changed when applying the two control objectives.

III. STABILIZING MODEL PREDICTIVE CONTROL DESIGN

In this section, a brief description of the NMPC scheme for nonholonomic robots is presented highlighting the necessary assumptions for terminal equality constraints stability proof. The general form of the nonlinear control system such as (1) can be expressed as follows:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \tag{8}$$

where $\mathbf{x}(t) \in R^n$ and $\mathbf{u}(t) \in R^m$ are the n dimensional state and m dimensional control vectors, respectively. The control objective is to compute an admissible control input $\mathbf{u}(t)$ to drive the system (8) to move toward the equilibrium point defined by ($\mathbf{x}_e(t) = 0$ and $\mathbf{u}_e(t) = 0$), where $\mathbf{x}_e$ and $\mathbf{u}_e$ are the differences between the system state vector and system control input and their reference values. The objective of the control algorithm is to minimize a weighted cost function given by [3]:

$$J(t, \mathbf{x}_e(\tau), \mathbf{u}_e(\tau)) = \int_t^{t+T} l(\tau, \mathbf{x}_e(\tau), \mathbf{u}_e(\tau)) d\tau \tag{9}$$

where $l(\tau, \mathbf{x}_e(\tau), \mathbf{u}_e(\tau)) = \mathbf{x}_e(\tau)^T Q \mathbf{x}_e(\tau) + \mathbf{u}_e(\tau)^T R \mathbf{u}_e(\tau)$ is referred to as the running cost, Q and R are positive definite symmetric weight matrices, and T is the prediction horizon. The stability of the predictive control has been shown to be guaranteed by imposing terminal equality constraints [15], [16]. Thus, at time t, the online-open-loop optimization problem of the NMPC controller can be outlined as the following:

$$\min_{\mathbf{u}} J(t, \mathbf{x}_e(\tau), \mathbf{u}_e(\tau)) \tag{10}$$

$$\begin{aligned} \text{subject to: } & \dot{\mathbf{x}}(\tau) = \mathbf{f}(\mathbf{x}(\tau), \mathbf{u}(\tau)) \\ & \mathbf{u}(\tau) \in U, (\tau \in [t, t+T]) \\ & \mathbf{x}_e(t+T) = 0 \end{aligned} \tag{11}$$

where $0 \in U \in R^m$ is a compact set specifying the control input saturation limits, and $\mathbf{x}_e(t+T) = 0$ defines the terminal state equality constraints. As outlined in [16], the controller stability can be proven if the following two assumptions are satisfied.

- The state vector $\mathbf{x}_r \in X$ is an equilibrium point for an admissible control value $\mathbf{u}_r \in U$, where $\mathbf{x}_\mathrm{r}$ is the reference state vector, and $X \in R^n$ is the state space set for the state vector $\mathbf{x}(t)$; this means that there exists a control value $\mathbf{u}_r \in U$ such that $\mathbf{f}(\mathbf{x}_r, \mathbf{u}_r) = \mathbf{x}_r$.
- The running cost function $l : X \times U \to R_0^+$ satisfies $l(\mathbf{x}_r, \mathbf{u}_r) = 0$ form $\mathbf{u}_r \in U$ obtained from the first assumption.

These assumptions can be easily checked by observing system (1) for the first assumption, and owing to the fact that the running cost function l has the quadratic form presented in (9), the second assumption is also satisfied.

IV. THE OPEN-SOURCE TOOLKIT

A software environment for automatic control and dynamic optimization (ACADO-Toolkit), which implements real-time NMPC routines, has been utilized to achieve the control purposes in this work. It is an open-source software package written in C++ that is completely self-contained; thus, coupling this package with external packages is optional [19]. Different studies implementing real-time MPC reported the use of dynamic optimization tool packages implementing NMPC algorithms such as Huge Quadratic Programming (HQP) package in [11], and nonlinear optimization (NLopt) package in [21]. However, as highlighted in these studies, a varying prediction horizon number of steps (N) was necessary for the real-time implementation [11], and a long update frequency is crucial to handle the long time-consuming online optimization step [21].

In order to be able to satisfy the final equality constraints presented in the previous section, it is crucial to conservatively maintain the number of the prediction horizon steps necessary for constraints satisfaction while maintaining a reasonably short controller update rate; thus, rendering the online optimization problem feasible. As will be presented in the next section, the used toolkit is allowing the application of such stability criteria in a real-time framework.

The general form of NMPC problem solved by (ACADO-Toolkit) is given as follows [19]:

$$\min_{x(.),u(.),p} \int_{t_0}^{t_0+T} \left\| h(x(t),u(t),p) - \eta(t) \right\|_Q^2 dt$$
$$+ \left\| m(x(t_0+T),p,t_0+T) - \mu \right\|_P^2$$
$$\text{subject to:}$$
$$x(t_0) = x_0 \tag{12}$$
$$\forall t \in [t_0, t_0+T] \quad 0 = f(t, x(t), \dot{x}(t), u(t), p)$$
$$\forall t \in [t_0, t_0+T] \quad 0 \geq s(t, x(t), u(t), p)$$
$$0 = r(x(t_0+T), p, t_0+T)$$

where $x(.)$ denote the system states, $u(.)$ the control input, p a time constant parameter (optional), T the prediction horizon time, $f(.)$ the model equation, $s(.)$ the path constraints, and $r(.)$ the terminal constraints. Moreover, the objective function is given in the least square form, where η and μ specify the tracking and terminal reference. It has to be indicated here that the running cost function $\left\| h(.) - \eta(t) \right\|_Q^2$ is solely considered in the work presented here. (i.e. $\left\| m(.) - \mu \right\|_P^2 = 0$). The control problem (12), written in scalar notation, can be easily matched with the robot control problem given by (10) and (11).

A C++ code which couples the control routines implemented by the used package, and Pioneer 3-AT robots' input files, has been developed; the developed code instantiates an NMPC execution routine each time step of the real experiments analyzed in the next section.

V. Experimental Results

This section presents the experimental results of the two control objectives. Pioneer 3-AT a research platform mobile robot shown in Fig.1 (b) is used in the conducted experiments. The robot is a 4-wheel skid steering mobile robot with an embedded computer, Ethernet-based communications, laser scanner, and other autonomous functions. For localization purpose, the robot uses the high accuracy 100 tick encoders with inertial correction for dead reckoning to compensate for skid steering. The computation of the control command vector **u** is performed on a 2.5 GHz quad-core machine with a Unix-based operating system and a memory of 6 GB. The calculated control signal is then transmitted over a wireless-based communication connection to a server running on the robot's onboard computer, whereas the robot state vector **x**(t) is fedback to the controller through the same connection. It has to be mentioned here that, as pioneer 3-AT is a skid steering mobile robot, model (1) is only an approximation of its motion [22].

The online optimization parameters including the sampling frequency, the number of prediction horizon steps (N), and the weight matrices Q and R given in (9), are chosen so that a satisfactory controller performance is achieved. In the following, the point stabilization results are presented first, then the trajectory tracking results.

A. Point Stabilization Results

To show the performance of the NMPC controller implemented for point stabilization, results of Pioneer 3-AT robot executing forward and parallel parking stabilization [6] are presented. In both stabilization cases, the robot starts from the initial pose $\mathbf{x}_0$ = $[0\ 0\ 0]^T$ (m, m, rad). For the forward parking case, the robot is commanded to stabilize at the pose $\mathbf{x}_r$ = $[2\ 2\ \theta_r]^T$, where θ_r has 4 different values starting at 0 (rad) and spaced by π/2 (rad). For the parallel parking case, the robot is commanded to stabilize at the pose $\mathbf{x}_r = [0\ 2\ \theta_r]^T$, where θ_r is as defined before. The controller update time step is chosen to be 0.3 seconds with number of prediction steps N = 30, leading to a prediction horizon time T = 9 seconds. The weight matrices Q and R of the cost function (9) are chosen as diagonal matrices with diagonal elements defined as (10, 10, 2) for Q, and as (20, 20) for R. In order to achieve accurate localization of the robot and satisfy its actuators saturation limits, the controller saturation limits, for the linear velocity v and angular velocity ω commands, are set as follows:

$$\begin{bmatrix} -0.25 \\ -0.7 \end{bmatrix} \leq \begin{bmatrix} v(m/s) \\ \omega(rad/s) \end{bmatrix} \leq \begin{bmatrix} 0.6 \\ 0.7 \end{bmatrix} \tag{13}$$

Results are summarized in Fig. 2 and Fig. 3. Fig. 2 (best viewed in colors) shows the trajectories exhibited by the robot when both forward parking and parallel parking stabilizations are performed. The robot position and orientation are represented by a circle with its center located at the robot position, and an arrow representing the robot orientation, respectively.

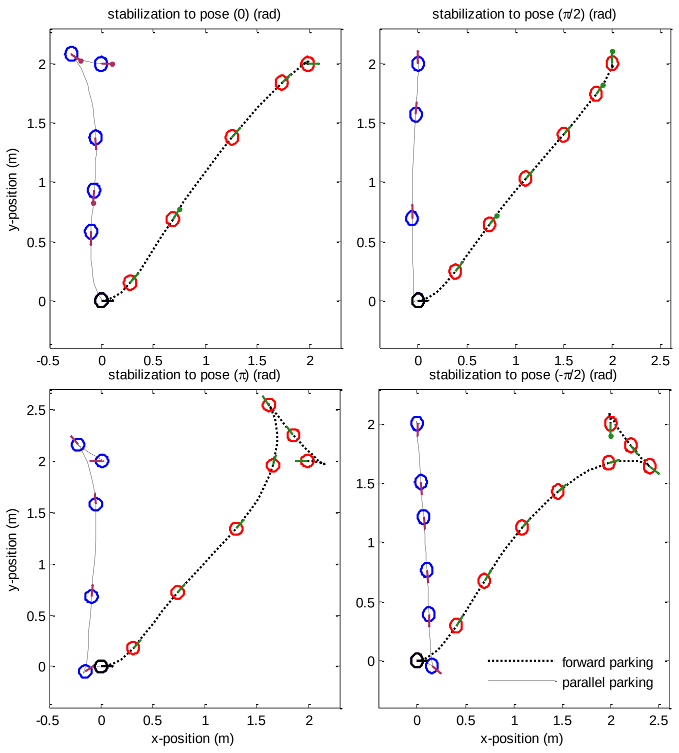

Fig. 2. Point-stabilization executed trajectories

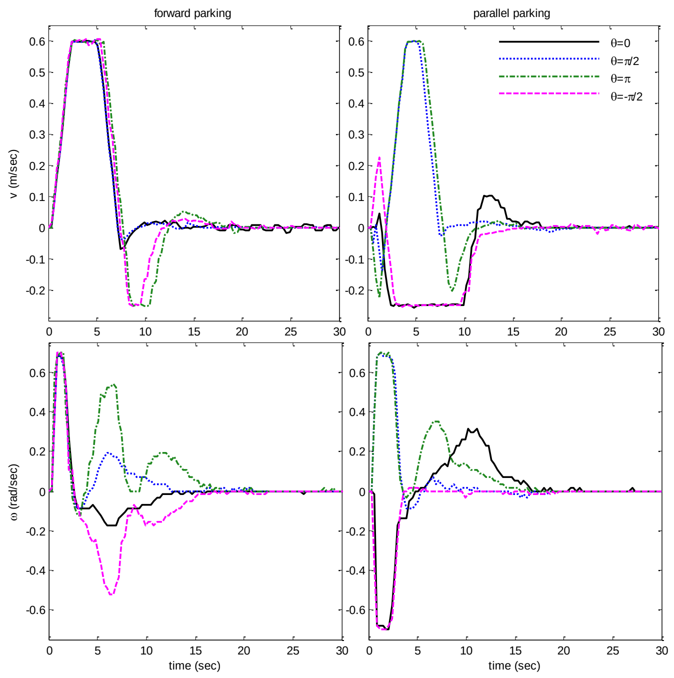

Fig. 3. Point-stabilization control signals

The robot starting position is highlighted with a black circle and a black arrow; a robot performing forward parking is represented by a red circle with a green arrow, whereas a robot performing parallel parking is represented by a blue circle with a brown arrow. As can be seen in the four sub-plots of Fig.2, the controller can stabilize the robot to the desired pose in all cases, by means of forward and backward motions. TABLE I summarizes the point stabilization performance parameters in terms of the error in the pose vector $\mathbf{x}_e$=[x_e (mm), y_e (mm), θ_e (rad)]^T, and the average computational cost per time step (milliseconds). The results show a satisfactory performance of the controller with a real-time demanding computational cost, for the point stabilization case. Fig.3 (best viewed in colors) shows that the actual linear and angular velocities of the robot are satisfying the saturation limits given by (13).

B. *Trajectory Tracking Results*

NMPC controller performance, for trajectory tracking, has been evaluated by considering two reference trajectories, namely, circular-shape (14), and eight-shape (15) trajectories. Parameters of the trajectories (14) and (15) are chosen such that the reference linear and angular velocities are not violating (13). Similar to the point stabilization case, the robot starts from the initial pose $\mathbf{x}_0$ = [0 0 0]^T (m, m, rad).

TABLE I. POINT STABILIZATION PERFORMANCE PARAMETERS

θ_r (rad)	Forward Parking				Parallel Parking			
	x_e	y_e	θ_e	*C*	x_e	y_e	θ_e	*C*
0	15	0	0	45	4	0	0	48
π/2	0	-4	0.02	44	0	-1	0	42
π	11	0	0.02	42	-9	0	0.02	40
-π/2	0	-7	-0.02	43	-1	-8	0	43

$$x_r(t) = 0.3 + 2\sin(0.25t)$$
$$y_r(t) = -2.3 + 2\cos(0.25t) \qquad (14)$$

$$x_r(t) = 0.3 + 1.5\sin(0.3t)$$
$$y_r(t) = 0.3 + 2.5\cos(0.15t) \qquad (15)$$

The controller update time step is chosen to be 0.2 seconds with number of prediction steps N = 30, leading to a prediction horizon time T = 6 seconds. The starting points of the reference trajectories (14) and (15) have been selected so that an initial error is imposed on the tracking problem. For the circular-shape tracking, the weight matrices Q and R of the cost function (10) are chosen as diagonal matrices with diagonal elements defined as (30, 30, 0.2) for Q, and as (50, 50) for R. For the eight-shape tracking, the diagonal elements of Q are (30, 30, 0.2), and the diagonal elements of R are (9, 5). Fig.4 and Fig.5 show the actual robot trajectories, error state vector $\mathbf{x}_e$, and linear and angular control signals, resulting from following the two reference trajectories, respectively. The robot start location is highlighted with a circle centered at the origin.

It can be seen from Fig.4 and Fig.5 that the elements of the error state vector $\mathbf{x}_e$ are asymptotically converging to the origin, whereas the robot speeds are converging to their reference values, despite the initial error they had .For the circular-shape tracking, the steady state value of the position error has been observed to be within (± 3.5 cm), and the orientation error within (± 0.04 rad), where the average computational cost per time step is (46 milliseconds) in this case. On the other hand, in the eight-shape tracking case, the steady state value of the position error has been observed to be within (± 4.5 cm), and the orientation error within (± 0.07 rad), where the average computational cost per time step is (37 milliseconds) in this case.

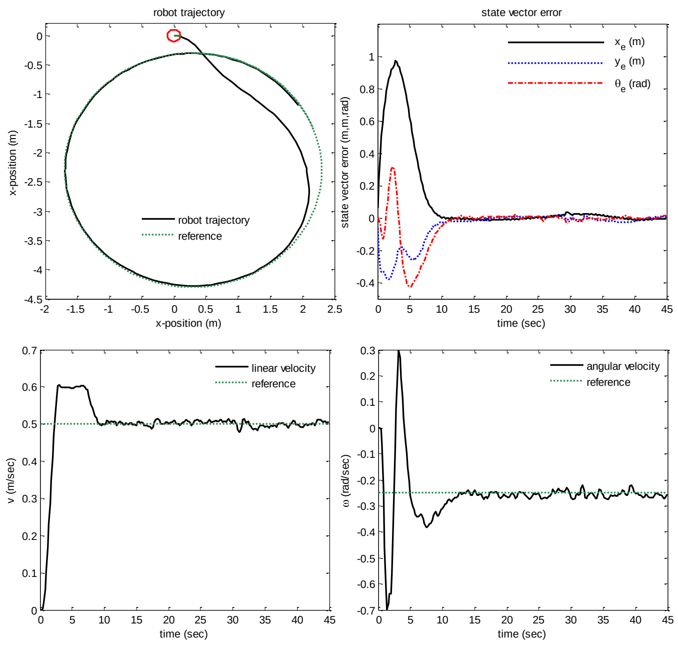

Fig. 4. Circular trajectory traking

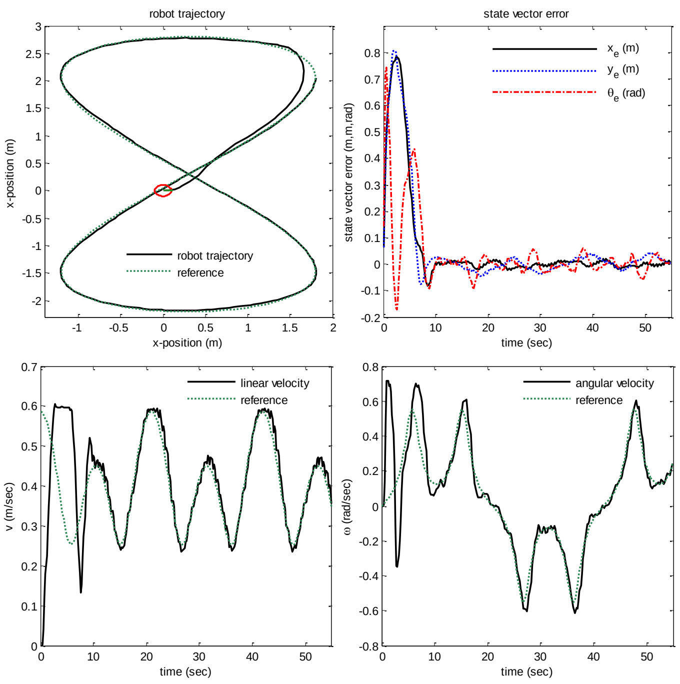

Fig. 5. eight-shape trajectory traking

Because the eight-shaped trajectory has sharp changes in its reference speeds, errors in the eight-shape trajectory tracking are relatively large when compared to circular-shape tracking. In both trajectory tracking cases, the control signals did not go beyond their limitations (13).

VI. Conclusion

In this paper, terminal equality constraints stabilizing NMPC has been proven to be applicable in the real-time context, for point stabilization and trajectory tracking control problems of mobile robots. The use of a toolkit implementing fast NMPC routines rendered, the computationally demanding stability criteria, tractable. Experiments were conducted using the Pioneer 3-AT research platform, after developing a C++ code that couples the used package and robot's input files. In the point stabilization case, the robot was commanded to perform forward and parallel parking with different orientations, whereas in the trajectory tracking problem, the robot was commanded to follow circular and eight-shape trajectories. The obtained results showed a satisfactory controller performance in terms of the point stabilization and tracking errors, and real-time requirements; hence, proving the applicability of the stabilizing NMPC scheme.

References

[1] J. P. Laumond, S. Sekhavat, F. Lamiraux, J. paul Laumond (editor, J. P. Laumond, S. Sekhavat, and F. Lamiraux, "Guidelines in nonholonomic motion planning for mobile robots," in Robot Motion Plannning and Control, pp. 1-53, Springer-Verlag, 1998.

[2] M. Michalek and K. Kozowski, "Vector-field-orientation feedback control method for a differentially driven vehicle," IEEE Transactions on Control Systems Technology, vol. 18, no. 1, pp. 45-65, 2010.

[3] D. Gu and H. Hu, "Receding horizon tracking control of wheeled mobile robots," IEEE Transactions on Control System Technology, vol. 14, no. 4, pp. 743-749, 2006.

[4] D. Gu and H. Hu, "A stabilizing receding horizon regulator for nonholonomic mobile robots," IEEE Transactions on Robotics, vol. 21, no. 5, pp. 1022-1028, 2005.

[5] W. E. Dixon, D. M. Dawson, F. Zhang, and E. Zergeroglu, "Global exponential tracking control of a mobile robot system via a PE condition," IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics, vol. 30, no. 1, pp. 129–142, Feb. 2000.

[6] G. Oriolo, A. De Luca, and M. Vendittelli, "WMR control via dynamic feedback linearization: Design, implementation, and experimental validation," IEEE Transactions on Control Systems Technology, vol. 10, no. 6, pp. 835–852, Nov. 2002.

[7] T. C. Lee, K. T. Sun, C. H. Lee, and C. C. Teng, "Tracking control of unicycle-modeled mobile robots using s saturation feedback controller," IEEE Transactions onControl Systems Technology, vol. 9, no. 2, pp. 305–318, Mar. 2001.

[8] G. Raffo, G. Gomes, J. Normey-Rico, C. Kelber, and L. Becker, "A predictive controller for autonomous vehicle path tracking," IEEE Transactions on Intelligent Transportation Systems, vol. 10, no. 1, pp. 92-102, 2009.

[9] P. Falcone, M. Tufo, F. Borrelli, J. Asgari, and H. Tsengz, "A linear time varying model predictive control approach to the integrated vehicle dynamics control problem in autonomous systems," 46th IEEE Conference on Decision and Control, pp. 2980-2985, 2007.

[10] G. Klanar and I. . krjanc, "Tracking-error model-based predictive control for mobile robots in real time," Robotics and Autonomous Systems, vol. 55, no. 6, pp. 460-469, 2007.

[11] J. Backman, T. Oksanen, and A. Visala, "Navigation system for agricultural machines: Nonlinear model predictive path tracking," Computers and Electronics in Agriculture, vol. 82, no. 0, pp. 32 - 43, 2012.

[12] H. Lim, Y. Kang, C. Kim, J. Kim, and B.-J. You, "Nonlinear model predictive controller design with obstacle avoidance for a mobile robot," IEEE/ASME International Conference on Mechtronic and Embedded Systems and Applications MESA 2008, pp. 494-499, 2008.

[13] F. Kuhne, W. Lages, and J. Gomes da Silva, J.M., "Point stabilization of mobile robots with nonlinear model predictive control," IEEE International Conference on Mechatronics and Automation, vol. 3, pp. 1163-1168, 2005.

[14] F. Xie and R. Fierro, "First-state contractive model predictive control of nonholonomic mobile robots," American Control Conference, pp. 3494-3499, 2008.

[15] J. B. Rawlings and K. R. Muske, "Stability of constrained receding horizon control," IEEE Transactions on Automatic Control, vol. 38, no. 10, pp. 1512–1516, Oct. 1993.

[16] L. Grüne, and J. Pannek, "Nonlinear Model Predictive Control: Theory and Algorithms", 1st ed, in Communications and Control Engineering, Springer-Verlag, 2011.

[17] Leineweber DB, Bauer I, Bock HG, and Schloder JP, "An efficient multiple shooting based reduced SQP strategy for large-scale dynamic process optimization. Part I: theoretical aspects," in Computers and Chemical Engineering,vol. 27, pp.157–166, 2003.

[18] Simon LL, Nagy ZK, Hungerbuehler K., "Swelling constrained control of an industrial batch reactor using a dedicated NMPC environment: OptCon," Nonlinear Model Predictive Control. Lecture Notes in Control and Information Sciences, vol. 384, pp. 531–539, Springer, 2009.

[19] B. Houska, H. Ferreau, and M. Diehl, "ACADO toolkit - An open source framework for automatic control and dynamic optimization," Optimal Control Applications and Methods, vol. 32, no. 3, pp. 298-312, 2011.

[20] A. M. Bloch, Nonholonomic Mechanics and Control, New York, NY: Springer, 2003.

[21] J. J. Park, C. Johnson, and B. Kuipers, "Robot navigation with model predictive equilibrium point control," International Conference on Intelligent Robots and Systems (IROS), pp. 4945-4952, 2012.

[22] H. Chung, L. Ojeda, and J. Borenstein, "Accurate mobile robot dead-reckoning with a precision-calibrated fiber-optic gyroscope," IEEE Transactions on Robotics and Automation, vol. 17, no. 1, pp. 80-84, 2001.