TRAJECTORY TRACKING FOR NONHOLONOMIC MOBILE ROBOTS USING PREDICTIVE METHODS*

Gustavo H. C. Oliveira†, José R. H. Carvalho‡

† *LAS/CCET/PUC-PR*
R. Imaculada Conceicao, 1155, Zip code: 80215-901, Curitiba-PR, Brazil
e-mail: oliv@ccet.pucpr.br
‡ *LRV/IA/CTI*
PO Box 6162. Zip code: 13083-970, Campinas-SP, Brazil
e-mail: regis@ia.cti.br

Keywords : predictive control, nonholonomic system, mobile robots, position control, path tracking.

Abstract

The present work is focused on the problem of closed loop position control and path tracking for nonholonomic unicycle-type type mobile robots. The main characteristic of the robot discussed in the paper is the presence of two coaxial wheels, each one driven by an independent DC-motor. These variables, i.e. the wheels speed, are then used to control the robot configuration and the application of Model Based Predictive Controllers (MBPC) in this context is proposed. In order to deal with the robot characteristics and improve its tracking behavior, some modifications in the classical MBPC strategy are presented. An actual nonholonomic robot platform, the *Khepera* robot, is used to validate the proposed strategy.

1 Introduction

Practical applications of mobile robots require the analysis of dynamic and kinematic modeling, sensing, spatial representation of the robot and its environment, control, navigation, and path planning. Mobile robots are, therefore, an excellent platform for application and validation of novel techniques. In this context, the low-level control of the robot movements is of primal importance since any upper level task to be performed depends directly on this robot's competence.

The present work addresses the problem of closed-loop position control and path tracking for a class of nonholonomic mobile robots. The main characteristic of the robot analyzed here is the presence of two coaxial wheels, each one driven by an independent DC motor. The speed of each wheel is the output of the controller algorithm. These variables are then used as set-points for PID controllers located in each DC motor. The controlled variables are the coefficients of the robot configuration vector, i.e. the coordinates x and y in the Euclidean space and the orientation angle θ. The system thus configured is multivariable, non-square, and non-linear. The state vector (x,y,θ) is measured in real-time through odometry and it is assumed that the classical hypothesis of movement without slippage in a flat and horizontal surface holds.

In the past decade, much attention has been given to the subject of path tracking and position control of nonholonomic mobile robots. One of the first works in this context is [1]. Later, in [2, 3], smooth and static (i.e., not explicitly dependent on the time variable) closed-loop control laws to solve the problem of tracking of a trajectory in Euclidean space are proposed, controlling both robot position and orientation. However, the trajectory, and hence the robot, must be always in movement. The same problem is solved in [4] using a dynamic feedback linearization technique. All these works point to a established result, presented in [5], which shows that there is no smooth and static closed-loop control law able to solve the problem of stopping a nonholonomic robot in a desired configuration (position and orientation).

On the other hand, in [6] it is presented a smooth non-static closed-loop control law which performs the stopping phase of the complete robot configuration, but the problem of implementing non-linear and non-static controllers remains an issue. In [7] it is proposed a solution based on attractor circumferences for the problem of stopping a nonholonomic unicycle-type robot in a given configuration, but the resulting control law is not always smooth. A non-smooth control law may result in extremely high acceleration signals which may not be followed by the robot actuators or can increase the odometry errors.

All the works cited above describe control design in the continuous-time domain. In these algorithms, it is not trivial to perform the tracking of a general trajectory with stopping phase. The method described in [7] deals with both problems, but there exist constraints over the trajectories that can be followed.

A very interesting solution for the closed loop control of mo-

*This work was partially sponsored under grants No. 97/13384-7 and 98/13562-5 by Fapesp, Brazil

bile robots is based on the well known Model Based Predictive Controller (MBPC) strategy [8]. The MBPC methods are often used in industrial applications [8] [9], included robotics [10, 11, 12, 13]. The work of [11] deals with an application, using a Labmate mobile robot, of neural networks for obstacle avoidance in a MBPC framework and [13] describes an application using a Nomad mobile robot.

The present paper reviews and extends the works [10, 12] where the use of MBPC controllers in the context of simulated nonholonomic robots is described. The main characteristics of algorithm presented here are: *i)* the real-time updating of the discrete model of the robot by using the odometry measurements; *ii)* the computing of the robot orientation (θ) set-point by using both the set-point for the x and y position, to guarantee the convergence to a desired final position and the stabilisation of the hole configuration, since the process is non square; finally, *iii)* the real-time tuning of the controller weighting based on the robot position, to improve the tracking. In order to validate the proposed strategy, an actual nonholonomic robot platform, the *Khepera* robot [14], is used in two different kinds of experiments. The first is a position control to a desired point in the Euclidean space and the second is the tracking, with stopping phase, of a trajectory which is build using circumference arcs and segment of straight lines, since it can be shown that the shortest path for nonholonomic robots is made by using this kind of trajectory [15].

The proposed strategy presents some interesting characteristics in the context of mobile robots, such as: *i)* it is based on a simple and well establish discrete time control design methodology; *ii)* the physical constraints in the actuators can be easily handled by the algorithm; *iii)* the predictive characteristic of the algorithm allows one to consider future values of the trajectory to improve the tracking; *iv)* it is able to perform both position control and tracking of a general trajectory with stopping phase.

This paper is organised as follows. In section 2, the kinematic model of the unicycle-type robot is reviewed and the robot's prediction equations are obtained. Then, in section 3, the modifications in the classical MBPC algorithm to the robot control are presented and, in section 4, the experiment results in the actual robotic platform are shown. Finally, the paper is concluded in section 5.

2 Kinematic Modeling

A complete description for kinematics models of mobile robots can be found in [4, 1] and, in this work, the model for the unicycle-type mobile robot is given by:

$$\begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \end{bmatrix} = \begin{bmatrix} \cos\theta(t) & 0 \\ \sin\theta(t) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v(t) \\ \dot{\theta}(t) \end{bmatrix} \tag{1}$$

where $x(\cdot)$, $y(\cdot)$ and $\theta(\cdot)$ denotes the robot configuration with respect to a fixed frame and $v(t)$ and $\dot{\theta}(t)$ are, respectively, the linear and angular robot velocities. The relation between these variables and the wheel speeds is given by:

$$\begin{bmatrix} v(t) \\ \dot{\theta}(t) \end{bmatrix} = \frac{r}{2} \begin{bmatrix} 1 & 1 \\ 1/R & -1/R \end{bmatrix} \begin{bmatrix} \dot{q}_1(t) \\ \dot{q}_2(t) \end{bmatrix} \tag{2}$$

where $\dot{q}_1(t)$ and $\dot{q}_2(t)$ are the angular speeds of left and right wheels, respectively. r is the length of the wheels' radii and R is the wheels' half axis length. The continuous-time model given by equations (1) and (2) can be rewritten as follows:

$$\dot{X}(t) = G(\theta(t))\, U(t) \tag{3}$$

where

$$X(t) = \begin{bmatrix} x(t) \\ y(t) \\ \theta(t) \end{bmatrix}, \quad U(t) = \begin{bmatrix} \dot{q}_1(t) \\ \dot{q}_2(t) \end{bmatrix} \tag{4}$$

$$G(\theta(t)) = \frac{r}{2} \begin{bmatrix} \cos\theta(t) & \cos\theta(t) \\ \sin\theta(t) & \sin\theta(t) \\ 1/R & -1/R \end{bmatrix} \tag{5}$$

Considering a fixed sampling time denoted by δt and assuming a zero order hold, it follows that:

$$X(k\delta t+\delta t) = X(k\delta t) + \Gamma(\theta(t))\, U(k\delta t)\,, \;\; k = \{k : k \in I\!N, k \geq 0\} \tag{6}$$

where

$$\Gamma(\theta(t)) = \int_{k\delta t}^{k\delta t+\delta t} G(\theta(t))dt \tag{7}$$

Since $\theta(t)$ can be represented as a linear combination of the input signal $U(k\delta t)$ components, then $\theta(t)$ is linear between two sample instants and the following equality holds:

$$\begin{aligned} \Gamma(\theta(t)) &= \int_0^{\delta t} G(\theta(k\delta t) + \dot{\theta}(k\delta t)t))dt \\ &= \frac{r}{2} \begin{bmatrix} \delta\sin(k\delta t) & \delta\sin(k\delta t) \\ -\delta\cos(k\delta t) & -\delta\cos(k\delta t) \\ \dfrac{\delta t}{R} & -\dfrac{\delta t}{R} \end{bmatrix} \end{aligned} \tag{8}$$

where

$$\delta\sin(k\delta t) = \frac{\sin[\,\theta(k\delta t + \dot{\theta}(k\delta t)\delta t)\,] - \sin[\theta(k\delta t)]}{\dot{\theta}(k\delta t)}$$
$$\delta\cos(k\delta t) = \frac{\cos[\,\theta(k\delta t + \dot{\theta}(k\delta t)\delta t)\,] - \cos[\theta(k\delta t)]}{\dot{\theta}(k\delta t)}$$

In this way, $\Gamma(\theta(t))$ can be written as $\Gamma(\theta(k\delta t), \dot{\theta}(k\delta t))$ or, in a more compact way, $\Gamma(k\delta t)$.

With this results, the j-step ahead prediction equation is derived. By using model (6) and by defining an operator Δ as $\Delta U(k) = U(k) - U(k-1)$ it follows[1]:

$$X(k+1) = X(k) + \Gamma(k)\Delta U(k) + \Gamma(k)U(k-1) \tag{9}$$

In this way, the j-step ahead prediction is given by:

$$\begin{aligned} X(k+j) &= \sum_{m=0}^{j-1} \sum_{i=m}^{j-1} \Gamma(k+i)\Delta U(k+m) + \\ &\quad + X(k) + \sum_{i=0}^{j-1} \Gamma(k+i)\, U(k-1) \end{aligned} \tag{10}$$

[1] In this paper, a normalized sampling time δt equal to 1 will be used.

From this model, it is direct remark that the parameters of the non-linear model can be updated on-line using the output measurements at the time instant k. Furthermore, it can be viewed as a linear model with on-line parameter adaptation.

3 Model-Based Predictive Control for the Unicycle-type Mobile Robot

Due to the nonholonomic constraint, the feedback control of the robot whole configuration is not trivial [4]. In this work, the use of a predictive method to control the robot position, following both a dynamic and static set-point signal is proposed.

Model based predictive control (MBPC) algorithms are nowadays used in many applications concerning industrial processes [8] and many variants of such a kind of controller have been proposed in the literature [16, 8, 17]. The MBPC methods are, by definition, based on predicting behavior of the process and the principle of the control law consists in minimizing a cost function over a future time horizon under certain constraints of the process. The cost function is defined in terms of tracking error, i.e., the difference between predicted output and desired set-point, where the output predictions are made by using a model of the process. A set of future control signals is obtained by the minimization of the cost function and only the first of these signals is applied in the process (receding horizon strategy).

Here, a model-based predictive control (MBPC) law for the unicycle-type robot is derived by minimizing a quadratic cost function as follows:

$$
\begin{array}{lr}
\Delta U^* \;=\; \operatorname{argmin} J(\Delta U) & \\
\text{subject to} & \\
\Delta\dot{q}_1(k+j-1)=0 \quad \forall \quad j>N_u & \\
\Delta\dot{q}_2(k+j-1)=0 \quad \forall \quad j>N_u & \\
-\dot{q}_{max}\le\dot{q}_1(k+j-1)\le\dot{q}_{max} \quad \forall\, j=1,\dots,N_u & (a.1)\\
-\dot{q}_{max}\le\dot{q}_2(k+j-1)\le\dot{q}_{max} \quad \forall\, j=1,\dots,N_u & (a.2)\\
-\Delta\dot{q}_{max}\le\Delta\dot{q}_1(k+j-1)\le\Delta\dot{q}_{max} \quad \forall\, j=1,\dots,N_u & (b.1)\\
-\Delta\dot{q}_{max}\le\Delta\dot{q}_2(k+j-1)\le\Delta\dot{q}_{max} \quad \forall\, j=1,\dots,N_u & (b.2)
\end{array}
\tag{11}
$$

where:

$\Delta U = [\,\Delta U(k)\cdots\Delta U(k+N_u-1)\,]^T$, with $\Delta U(k+j) = [\,\Delta\dot{q}_1(k+j),\,\Delta\dot{q}_2(k+j)\,]^T$ and $\Delta\dot{q}_i(k)=\dot{q}_i(k)-\dot{q}_i(k-1)$;

$(a.1)$ and $(a.2)$ (resp. $(b.1)$ and $(b.2)$) are the constrains for the wheels speeds (resp. accelerations), where $\dot{q}_{max}$ ($\Delta\dot{q}_{max}$) is the maximum value allowed for $|\,\dot{q}_i(\cdot)\,|$ (resp. $|\Delta\dot{q}_i(k)|$).

$J(\Delta U)$ is the controller cost function, which is defined as the following quadratic function:

$$
\begin{aligned}
J= & \sum_{j=N_1}^{N_y}\left[X^d(k+j)-X(k)\right]^T \Xi\left[X^d(k+j)-X(k)\right] \\
& +\sum_{j=1}^{N_u} U^T(k+j-1)\Lambda U(k+j-1)
\end{aligned}
\tag{12}
$$

In this equation:

N_1 and N_y define the output prediction horizon and N_u is the control horizon.

Ξ and Λ are the weighting matrices, acting respectively on the output signal errors and the control signal.

$X^d(k) = [\,x^d(k),\,y^d(k),\,\theta^d(k)\,]^T$ is the set-point vector;

$X(k) = [\,x(k),\,y(k),\,\theta(k)\,]^T$ is the state vector;

$X(k+j)$ is the j-step ahead prediction of the robot output, which is computed using equation (10).

The optimal wheels' speed $U(k)=[\,\dot{q}_1(k),\,\dot{q}_2(k)\,]^T$, applied in the process, is obtained from the optimal ΔU^* vector computed at k, as $U(k)$=$U(k-1)+\Delta U^*(k)$. In the next sampling time, all the procedure is repeated, following the classical receding horizon control strategy.

The cost function (12) can be easily rewritten as a quadratic function in relation to the vector ΔU and then the optimization problem (11) is given by:

$$
\begin{array}{c}
\min\limits_{\Delta u} \quad \Delta U^T Q\Delta U + f^T\Delta U \\
\text{subject to} \\
\mathcal{A}\;\Delta U \;\le\; v
\end{array}
\tag{13}
$$

where Q and f are appropriated matrices, $\mathcal{A}$ and v are built using the information about the process constraints presented in equation (11). Thus, a single Quadratic Programming (QP) problem relative to the vector ΔU must be solved to obtain the control law.

3.1 Improving the system convergence

The unicycle-type robot is a non-square system, i.e., it has two input and three output signals. Thus, there are more degrees of freedom than manipulated variables and the NLMBPC method is not able to lead the robot states to a pre-defined configuration.

Here, it is proposed to solve this problem by rewriting the orientation set-point as a function of the position set-point, as follows: the third component of the set-point vector $X^d(k)$, $\theta^d(k)$, is computed as the angle (in the range $-\pi$ and π) between the x-axis and the segment that links the current robot position to the position set-point, i.e.:

$$
\theta^d(k+1) = \operatorname{atan}\left(\frac{x^d(k)-x(k)}{y^d(k)-y(k)}\right)
\tag{14}
$$

In this way, one degree of freedom is eliminated since $\theta^d(k)$ is no more an independent variable, but it is a function of $x^d(k)$ and $y^d(k)$. This time-varying orientation set-point always tends to turn the robot in such a way that it follows the minimal path that leads to the desired position, assuring the system convergence.

3.2 Improving the tracking: tuning of the weighting matrix Ξ

In this section, a methodology for the real-time updating of the Ξ matrix components is presented. Let equation (12) be the MBPC cost function, where ξ_x, ξ_y and ξ_θ are the coefficients of Ξ that correspond to the position and orientation weights.

The tracking of the robot between two specific points can be improved, in the sense of giving a more rectilinear movement, by following the heuristics for real-time tuning of Ξ matrix given bellow:

1. *When the robot is far from the end point, ξ_θ has to be higher. Then, the robots tends to keep its trajectory in a straight line.*

2. *When the robot near the end point, but the orientation erros keeps high, then ξ_θ also has to be high.*

3. *When the robot is near the end point with a small orientation error, then ξ_θ also has to be small to reinforce the correction of the position error.*

This rules were obtained from observations of the robot behavior executing an approach to a desired position. The first two rules make the MBPC algorithm to prioritize the correction of robot orientation, while the third rule deal with the correction of the robot position. Since this rules are described as "high", "small", "near", "far", i.e. subjective qualifiers, fuzzy systems [18] are the natural approach to implement rules above. However, it is easy to note that this rules may also be approximated by the following mathematical operator:

$$\xi_\theta(k) = exp\left(k_d e_d + k_\theta e_\theta\right) \tag{15}$$

where e_d is the Euclean distance between the robot and the end point at instant k, e_θ is the orientation error with respect to a given reference $w_\theta(k)e_{max}$ and e_{max} is a pre-defined maximum value for sum of the errors.

Preliminary simulations shown that both fuzzy based system and the mathematical function above are efficients in real-time tuning $\xi_\theta(k)$, but the equation (15) has a much more simple implementation.

4 Experimental Results

In this section, the proposed MBPC method is applied to control a unicycle-type mobile robot in both position and path tracking with stopping phase.

4.1 Experimental Setup

Figure 2 shows both the hardware and software setup used in all experiments.

Computer Platform: Sun UltraSparc 1 workstation, 167 MHz. Represented by the bold box in Figure 2;

Robot Platform: The mobile robot *Khepera* (Figure 1). The *Khepera* robot, which is shown in Figure 1, has wheel radius $r = 0.65\ cm$ and wheels' half axis length $R = 2.5\ cm$. The speed of each wheel is given by a multiple of a minimal speed, i.e.:

$$\dot{q}_i(t) = k_i(t)\dot{q}_m \tag{16}$$

where $\dot{q}_m = 0.8125\ rad/s$, $i \in \{1,2\}$ and $k_i(t)$ is an integer with $k_i(t) \in \{-127, \ldots, 127\}$ [14]. Thus, before being sent to the robot, the control signal $U(k)$ obtained is adjusted to match equation (16).

Predictive control: Implemented using MATLAB functions and scripting language. Represented by the box labeled *controller* in Figure 2;

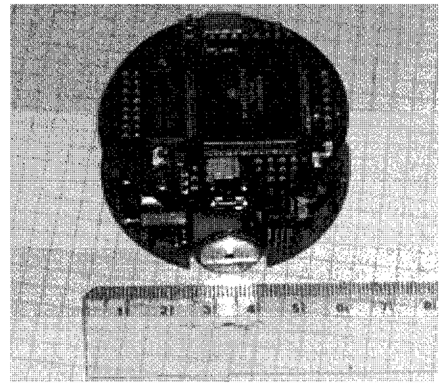

Figure 1: The *Khepera* robot.

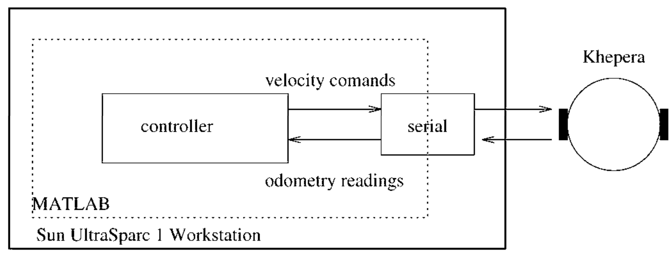

Figure 2: Setup used in all experiments.

Serial Communication: Implemented in C language and encapsulated as MATLAB functions using C-MEX facilities. Due to this hybrid nature, the box labeled *serial* (which represents this module in Figure 2) is not completely inside the dotted box representing MATLAB.

4.2 Position control

The robot starts from $X(0) = [0,\ 0,\ 0]^T$ and the experiment is performed until the robot reaches the final position, given by the point $X^d = [30cm,\ 30cm]^T$, with a pre-specified precision. The state vector of the robot is updated via its odometry and the sampling time is 0.1 seconds. The controller reference signal is a sequence of points equal to X^d.

The tuning parameters of the MBPC algorithm are selected as follows: $N_1 = 1$, $N_y = 3$ and $N_u = 1$. The control signal weighting Λ is set equal to a matrix of zeros in the $\Re^{2\times 2}$ space. The weighting matrix Ξ is initialized equal to the identity matrix. $\xi_\theta(k)$ is a time varying variable which follows equation (15), where e_{max} is set equal to 40. The speed of the robot is bounded in the control algorithm at 12.31 rad/s, corresponding to 10 $pulses/10msec$ using the *Khepera*'s speed unit. The acceleration constraint allows the speed to grow at most 6.15 rad/s (5 $pulses/10msec$) during two consecutive sample times. This constraints are conservatives in relation to the actual robot capability in order to decrease the influence of odometry errors.

Figure 3 shows the robot trajectory (solid line), which is interpolated through the values measured by the robot odometry (circles) at each sampling time. The speed of the robot wheels, computed by the control algorithm, are shown in Figure 4.2.

This experiment shows that, with the proposed methodology, the *Khepera* is able to reach a pre-defined position in the Euclidean space.

Figure 5 shows the same experiment but with a time-invariant value for Ξ matrix, given by $\xi_x = 1$, $\xi_y = 1$ and $\xi_\theta = 4$. By comparing this experiment with the previous one, it can be noted that the methodology for on-line tuning of Ξ can effectively improve the robot behavior when moving towards the set-point.

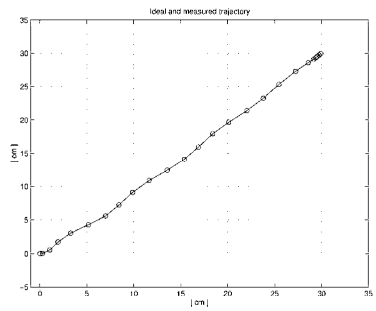

Figure 3: The *Khepera* reaching a position in the Euclidean space. The robot trajectory (solid line), and odometry measurements (circles).

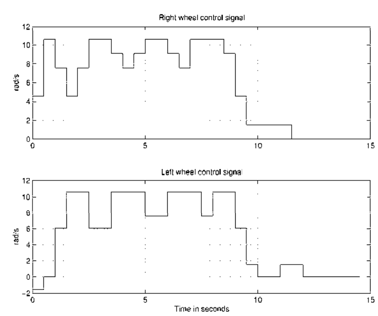

Figure 4: The speed sent to each wheel to reach a desired position in the Euclidean space.

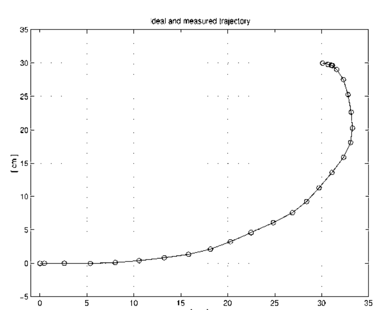

Figure 5: The *Khepera* reaching a position using a time invariant Ξ. The robot trajectory (solid line), and odometry measurements (circles).

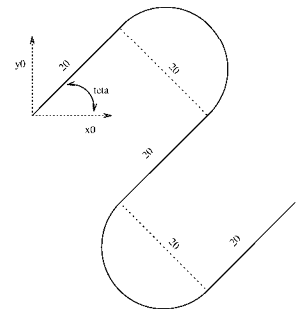

Figure 6: The trajectory to be followed by the *Khepera* robot.

4.3 Path tracking with stopping phase

The trajectory to be followed in this experiment is an S composed by three segments and two arcs of circumference (see Figure 6). The robot starts from $X(0) = [0,\ 0,\ 0]^T$ and the reference trajectory starts from the point [0, 0], with orientation $\pi/4$. The state of the robot is updated via its odometry with sampling time equal to 0.5 seconds. The MBPC parameters are the same as in the previous experiment with a time varying update of Ξ matrix.

Figure 7 shows the reference trajectory (solid line) and the robot position sensed by the robot odometry (circles) at each sampling time. The speed of the robot wheels to perform this experiment are shown in Figure 4.3. These figures shows that the tracking errors remains within an acceptable bound and the robot is able to perform the stopping phase. These was achieved using a simple control law that do not allows high changes in the control signal which could degenerate the tracking due to an increasing of odometry errors.

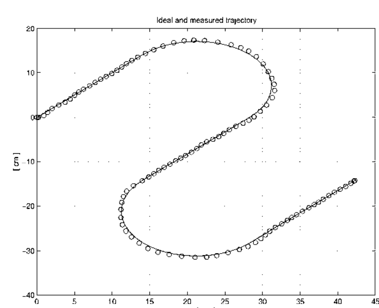

Figure 7: The *Khepera* tracking a trajectory in the Euclidean space. The reference trajectory (solid line), and odometry measurements (circles).

5 Conclusions

In this work, the problem of closed-loop position control and path tracking for nonholonomic unicycle-type mobile robot was addressed. This type of robot is a non-linear, non-square and time-varying process.

In order to deal with the robot characteristics, the utilization of MBPC-type control algorithms was proposed. The main

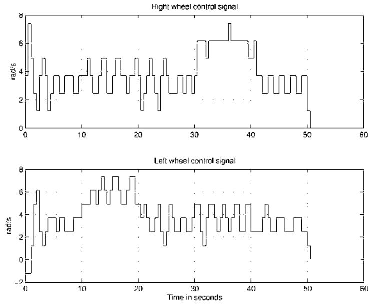

Figure 8: The speed sent to each wheel to perform the path tracking.

characteristics of algorithm were: a real-time updating of the robot model by using measurements, a guideline to computing of the robot orientation (θ) set-point and a strategy for real-time tuning of the controller weighting. The authors have submitted an extended version of this work [19], including a comparison with the non-linear solution and a discussion about stability. In order to validate the proposed strategy, the *Khepera* was used to solve two specific problems: the position control and the tracking of a pre-defined trajectory. It was shown that the proposed MBPC algorithm is effective in both cases, providing a quite good behavior during the tracking of a pre-defined trajectory with stopping phase and/or during the approach to a desired position in the Euclidean space. Furthermore, the method is based on a simple control law which can deal with acctuators constraints and consider future instants of the path.

References

[1] C. Samson and K. Ait-Abderrahim. Mobile robot control part 1 : Feedback control of a nonholonomic wheeled cart in cartesian space. Technical Report RR 1288, INRIA, Sophia-Antipolis, Octobre 1990.

[2] Claude Samson. Velocity and torque feedback control of a nonholonomic wheeled cart in cartesian space. In *Int. Workshop in Adaptative and Nonlinear Control: Issues in Robotics*, Grenoble, France, 1990.

[3] Claude Samson. Time-varying feedback stabilization of noholonomic car-like mobile robots. Technical report, INRIA, Centre de Sophia-Antipolis, September 1991.

[4] B. d'Andrea Novel, G. Campion, and G. Bastin. Control of nonholonomic wheeled mobile robots by state feedback linearization. *The International Journal of Robotics Research*, 14(6):543–559, December 1995.

[5] R. W. Brockett. Asymptotic stability and feedback stabilization. In *Proc of the conf. held at Michigan Technological University*, Michigan, USA, June-July 1983.

[6] Claude Samson. Control of chained systems: Application to path following and time-varying point-stabilization of wheeled vehicles. *IEEE Transactions on Automatic Control*, 40(1):64–77, January 1995.

[7] Ole Jakob Sørdalen. *Feedback Control of Nonholonomic Mobile Robots*. PhD thesis, The Norwegian Institute of Technology, Trondheim, Norway, 1993.

[8] D. W. Clarke. *Advances in Model Based Predictive Control*. Oxford University Press, 1994.

[9] J. Richalet. Industrial applications of model based predictive control. *Automatica*, 29(5):1251–1274, 1993.

[10] G. H. C. Oliveira and J. F. Lopez. Commande pr'dictive appliqu'e aux robots mobiles. Technical Report no. 95-35, Laboratoire de Signaux et Systémes de Sophia Antipolis - I3S, Nice/France, August 1995. in french.

[11] J. G. Ortega and E. F. Camacho. Neural predictive control for mobile robot navigation in a partially structured static environment. In *Proc. of 13th IFAC World Congress*, volume Q, pages 429–434, San Francisco/USA, 1996.

[12] G. H. C. Oliveira, J. R. H. Carvalho, J. F. Lopez, R. M. Nazzetta, and G. Favier. Controle de posição de robôs tipo cart via abordagem preditiva. In $8^{o.}$ *Latin American Congress of Automaic Control*, Viña del Mar/Chile, 1998. to be published.

[13] J. Normey-Rico, J. Gómez-Ortega, I. Alcalá-Torrego, and E. F. Camacho. Nlow time-consuming implementation of predictive path-tracking control for a "syncro-drive" mobile robot. In *Proceedings of the AMC98*, Coimbra, Pt, June 1998.

[14] K-Team. *Khepera - users manual*. Laboratoire de microinformatique - Swiss Federal Institute of Technology (EPFL), INF - Ecublens, 1015 Lausanne/Switzerland, 1993.

[15] J. A. Reeds and R. A. Shepp. Optimal paths for a car that goes both forward and backward. *Pacific Journal of Mathematics*, 145(2):367–393, 1990.

[16] R. Soeterboek. *Predictive Control : A Unified Approach*. Prentice Hall International, 1992.

[17] G. H. C. Oliveira, J. F. Lopez, G. Favier, and W. C. Amaral. Review of predictive control methods using laguerre functions. In *Proc. of IEEE/IMACS Multiconference on Computational Engineering in Systems Applications - CESA, Symposium on Control Optimization and Supervision*, volume 1, pages 304–309, Lille/France, 1996.

[18] W. Pedrycz. *Fuzzy Control and Fuzzy Systems*. Research Studies Press - Wiley & Sons, 2nd edition, 1993.

[19] G. H. C. Oliveira and J. R. H. Carvalhor. Closed-loop control of nonholonomic mobile robots using predictive methods. *European Journal Of Control*. submitted.