Experimental Comparison of Trajectory Tracking Algorithms for Nonholonomic Mobile Robots

Mišel Brezak, Ivan Petrović and Nedjeljko Perić

University of Zagreb
Faculty of Electrical Engineering and Computing
Zagreb, Croatia
Email: {misel.brezak, ivan.petrovic, nedjeljko.peric}@fer.hr

***Abstract*—This paper presents an experimental overview of common trajectory tracking methods for nonholonomic mobile robots: linear control, nonlinear control and model predictive control. All methods are compared experimentally on a two-wheel mobile robot with differential drive. The goal was to determine which control method is the best with respect to robustness and low trajectory tracking error. Thereby, a special emphasis is given on a fast real-time trajectory tracking and behavior in conditions near robot velocity and acceleration limits.**

I. Introduction

The field of mobile robotics is an active research area with promising new application domains in industrial as well as in service robotics. Mobile robots are especially appropriate in applications where flexible motion planning is required. There are many robot navigation strategies, and in cases where operating environment map is known, approach with trajectory planning algorithms is commonly used. Here the term trajectory denotes the path that robot should traverse as a function of time. Trajectory planner generates the appropriate trajectory with goal of arriving at a particular location, patrolling trough specified area etc, and at the same time avoiding collisions with different kinds of obstacles. To obtain feasible trajectory, that is, the trajectory that is robot actually able to track, the planner also needs to consider various robot physical and dynamic limitations such as its velocity and acceleration limits. A trajectory can be generated in real-time on the basis of current sensor readings or generated in advance on the basis of operating environment map.

Once the trajectory is planned, the robot must actually track it. This means that reference point on the robot must follow the planned trajectory. In other words, the distance between the reference point on the robot and current path reference point must be kept as small as possible. Ideally, this could be achieved using only feedforward commands. But of course, measurement errors and other real world issues force us to use feedback control and combine it with feedforward commands.

An experimental overview of trajectory tracking controllers based on linear, nonlinear and dynamic feedback linearization methods is given in [1]. However, advanced control methods, such as model predictive control, are not considered. Some examples of using model predictive controller for trajectory tracking of nonholonomic systems can be found in [2] and [3], and a newer example is in [4]. As experimental comparison of denoted control approaches is not covered in literature, the main contribution of this paper is to give experimental testing and comparison of denoted control approaches, especially in conditions near robot velocity and acceleration limits when high-speed real-time control is required.

This paper is organized as follows. In section 2 a robot model is described. In section 3 the compared trajectory tracking controllers are presented. In section 4 experimental results are presented. The paper ends with conclusions and ideas for future work.

II. Robot Model

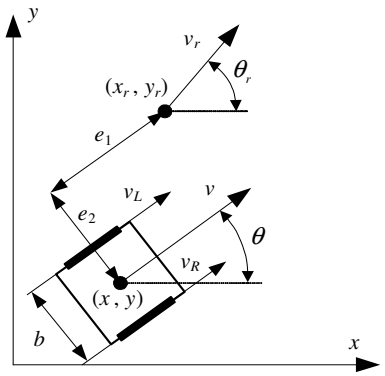

Fig. 1. Robot model

Kinematics of a mobile robot with differential drive is defined by a Jacobian matrix $J(\theta)$ that transforms tangential and angular velocities expressed in coordinates of mobile robot base to its velocities expressed in global Cartesian coordinates (Fig. 1):

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = J \begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}. \qquad (1)$$

Angle θ is robot heading direction that is taken counter-clockwise from the x-axis. The mapping between tangential (driving) and angular (steering) velocities and circumferential

velocities of the wheels (see Fig. 1), which are actual commands to the system, is given by:

$$v_R = v + \omega b/2, \quad v_L = v - \omega b/2, \tag{2}$$

where b is the distance along the axle between the centers of the drive wheels, and v_R and v_L are circumferential velocities of the right and left wheel, respectively.

Also, the nonholonomic constraint that the driving wheels purely roll and do not slip must be fulfilled, which is:

$$-\dot{x}\sin\theta + \dot{y}\cos\theta = 0. \tag{3}$$

It is important to mention that dynamics of the robot is not considered in kinematic model. Therefore, it is assumed that robot exactly realizes velocity commands v and ω. Of course, due to robot and actuator dynamics, and also nonidealities such as friction, gear backslash, wheel slippage, actuator deadzone and saturation, the robot cannot exactly realize velocity commands. Nevertheless, in case when reference trajectory has continuous profile with velocities and accelerations that do not exceed robot limitations, robot can approximatively track the reference velocity commands, i.e. the trajectory is feasible for the robot.

But, in case when trajectory tracking controller temporarily generates command velocities higher than actual robot limitations, a velocity saturation will occur. During this time it is important that velocity commands are saturated so that robot retains its steering direction, i.e. path curvature. This means that ratio ω/v (which is equal to path curvature) has to be preserved. Therefore, if v_{max} and ω_{max} are maximum robot's tangential and angular velocity commands, respectively, bounded command velocities v_c and ω_c that preserve path curvature are obtained by first defining

$$\sigma = \max\left\{|v|/v_{\max}, \quad |\omega|/\omega_{\max}, \quad 1\right\}.$$

Then the following algorithm is used to determine bounded velocities [5]:

$$\begin{gathered} v_c = v, \quad \omega_c = \omega, \quad \text{if } \sigma = 1; \\ v_c = \operatorname{sign}(v) v_{max}, \quad \omega = \omega/\sigma, \quad \text{if } \sigma = |v|/v_{max}; \\ v_c = v/\sigma, \quad \omega = \operatorname{sign}(\omega)\omega_{max}, \quad \text{else.} \end{gathered} \tag{4}$$

III. Trajectory Tracking Controllers

Trajectory tracking problem is actually a problem of computing appropriate robot commands so that robot (i.e. reference point on the robot) tracks a reference point of the trajectory. Here the trajectory is given as a desired robot state as a function of time:

$$\left[x_r(t), y_r(t), \theta_r(t), v_r(t), \omega_r(t)\right]^T, \tag{5}$$

where $(x_r(t), y_r(t))$ is reference position in cartesian space, $\theta_r(t)$ is reference robot orientation, and $v_r(t)$ and $\omega_r(t)$ are reference tangential and angular velocities, respectively. Of course, trajectory must be planned so that the nonholonomic constraint (3) is fulfilled. If the time derivatives $\dot{x}_r$, $\dot{y}_r$, $\ddot{x}_r$ and $\ddot{y}_r$ exist, i.e. if desired Cartesian motion $(x_r(t), y_r(t))$ is twice differentiable, state variables $\theta(t)$, $v_r(t)$ and $\omega_r(t)$ in trajectory (5) are not independent, but can be calculated as

$$\theta_r(t) = \operatorname{arctan2}(\dot{y}_r(t), \dot{x}_r(t)) \tag{6}$$

$$v_r(t) = \pm\sqrt{\dot{x}_r^2(t) + \dot{y}_r^2(t)} \tag{7}$$

$$\omega_r(t) = \frac{\dot{x}_r(t)\ddot{y}_r^2(t) - \dot{y}_r^2(t)\ddot{x}_r(t)}{\dot{x}_r^2(t) + \dot{y}_r^2(t)}. \tag{8}$$

But if for some time t the reference tangential velocity $v_r(t)$ is zero, neither the reference angular velocity $\omega_r(t)$ nor the reference angle $\theta_r(t)$ are defined by eq. (6) and eq. (8), and must be given explicitly.

Commonly, the trajectory tracking controller consists of feedforward and feedback part. The values of feedforward commands are computed based on reference tangential and angular velocity (v_r and ω_r) in trajectory (5). The output of feedback part is then computed based on current tracking error, and is superposed on generated feedforward commands. Therefore robot velocity command vector $u = [v \;\; \omega]^T$ can be written as

$$u = u_F + u_B, \tag{9}$$

where u_F is feedforward, and u_B is feedback part. Feedforward component is obtained using nonlinear transformation of reference velocities

$$u_F = \left[v_r cos(e_3) \quad \omega_r\right]^T, \tag{10}$$

and is equal for all presented controller schemes, while the feedback part is unique for each particular controller.f The structure of general trajectory tracking controller, which consists of feedforward and feedback part, is shown in Fig. 2.

A. Linear Controller

The simplest design of trajectory tracking controller is obtained using tangent linearization along the reference trajectory. Here it is required that robot states track given reference trajectory, so that dynamics of tracking error has to be stabilized. The tracking error $e(t)$ is defined as follows

$$e(t) = \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r - x \\ y_r - y \\ \theta_r - \theta \end{bmatrix}. \tag{11}$$

Note that errors e_1 and e_2 are actually position error components expressed in robot local coordinate system, as shown in Fig. 1. By deriving equation (11) and taking into account robot kinematics (1) and (3) the error dynamics becomes

$$\begin{bmatrix} \dot{e}_1 \\ \dot{e}_2 \\ \dot{e}_3 \end{bmatrix} = \begin{bmatrix} \cos e_3 & 0 \\ \sin e_3 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_r \\ \omega_r \end{bmatrix} + \begin{bmatrix} -1 & e_2 \\ 0 & -e_1 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}, \tag{12}$$

Rewriting eq. (12) and using equations (9) and (10), the following error dynamics model is obtained

$$\dot{e} = \begin{bmatrix} 0 & \omega & 0 \\ -\omega & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} e + \begin{bmatrix} 0 \\ \sin e_3 \\ 0 \end{bmatrix} v_r + \begin{bmatrix} -1 & 0 \\ 0 & 0 \\ 0 & -1 \end{bmatrix} u_B. \tag{13}$$

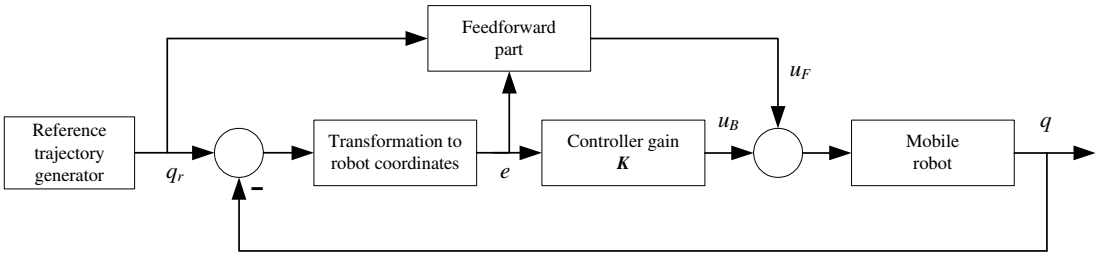

Fig. 2. Trajectory tracking controller structure

Linearizing eq. (13) around the reference trajectory ($e = 0$, $u_B = 0$, $v = v_r$, $\omega = \omega_r$) gives a linear model

$$\dot{e} = \begin{bmatrix} 0 & \omega_r & 0 \\ -\omega_r & 0 & v_r \\ 0 & 0 & 0 \end{bmatrix} e + \begin{bmatrix} -1 & 0 \\ 0 & 0 \\ 0 & -1 \end{bmatrix} u_B. \quad (14)$$

It can be shown that this system can be locally stabilized around reference trajectory which consists of linear or circular paths with constant velocity [1]. Furthermore, it is shown that by using linear design methods it is possible to locally stabilize given system even for arbitrary feasible trajectories under condition that they do not come to a stop.

Now, following [1], the design procedure of the linear controller will be shortly described. First, the linear feedback law is defined as

$$u_B(t) = K_s e(t), \quad (15)$$

where the gain matrix K_s is defined as

$$K_s = \begin{bmatrix} k_1 & 0 & 0 \\ 0 & \text{sign}(v_r)k_2 & k_3 \end{bmatrix}. \quad (16)$$

The structure of gain matrix (16) can be intuitively explained by looking at Fig. 1. The tangential error e_1 is reduced by manipulating the command tangential velocity v_B via gain k_1. The orientation error e_3 is reduced similarly by controlling command angular velocity ω_B of the robot via gain factor k_3. Finally, the orthogonal error e_2 is also reduced by changing the angular velocity, but here the driving direction (forward or backward) should also be taken into account.

Now the gains k_1, k_2, k_3 are determined by using desired closed-loop characteristic equation

$$(\lambda + 2\zeta\omega_n)(\lambda^2 + 2\zeta\omega_n\lambda + \omega_n^2) \quad (17)$$

with one negative eigenvalue at $-2\zeta\omega_n$ and a complex pair with natural angular frequency $\omega_n > 0$ and damping coefficient $\zeta \in (0, 1)$. The closed loop characteristic (17) is obtained by choosing the gains

$$k_1 = k_3 = 2\zeta\omega_n, \;\; k_2 = \frac{\omega_n^2 - \omega_r(t)^2}{|v_r(t)|}. \quad (18)$$

The problem with this choice of gains is that k_2 goes to infinity as reference linear velocity v_r goes to zero. The possible gain scheduling scheme that solves the problem is obtained by letting $\omega_n(t) = \sqrt{\omega_r^2(t) + g v_r^2(t)}$ [1], which gives:

$$k_1 = k_3 = 2\zeta\sqrt{\omega_r^2(t) + g v_r^2(t)}, \quad k_2 = g|v_r(t)|, \quad (19)$$

where parameter $g > 0$ gives an additional degree of freedom. It can be noted that with this choice of gains linear controller actually becomes nonlinear, but we still call it linear as it is obtained using linear design methods. This is also a continuous controller, but under assumption that sampling period is small, it can also be used as discrete controller.

B. Nonlinear Controller

The main importance of nonlinear controller design is that its global asymptotic stability can be proven. First of all, the feedback control is defined with following gain matrix [6]

$$K_n = \begin{bmatrix} k_1(v_r(t), \omega_r(t)) & 0 & 0 \\ 0 & \overline{k_2} v_r(t) \frac{sin(e_3)}{e_3} & k_3(v_r(t), \omega_r(t)) \end{bmatrix}, \quad (20)$$

where $\overline{k_2} > 0$ is constant positive gain, and $k_1(\cdot,\cdot)$ and $k_3(\cdot,\cdot)$ are positive continuous gain functions. Assuming that v_r and ω_r are bounded and with bounded derivatives, and that $v_r(t) \nrightarrow 0$ and $\omega_r(t) \nrightarrow 0$ when $t \rightarrow \infty$, using Lyapunov analysis one can prove that the control law (20) globally asymptotically stabilizes the origin $e = 0$.

Following the design of previous linear controller, the gain functions k_1 and k_3, and the constant gain $\overline{k}_2$ can be chosen as

$$k_1(v_r(t), \omega_r(t)) = k_3(v_r(t), \omega_r(t)) = 2\zeta\sqrt{\omega_r^2(t) + g v_r^2(t)}, \quad \overline{k}_2 = g, \quad (21)$$

where $g > 0$ and $\zeta \in (0, 1)$.

C. Model Predictive Controller

The main idea of model predictive controller (MPC) approach is to minimize the difference between reference trajectory tracking error and predicted trajectory tracking error, as well as control effort. Accordingly, a quadratic cost function can be written as

$$J(u_B, k) = \sum_{i=1}^{h} \epsilon^T(k,i)\, Q\, \epsilon(k,i) + u_B^T(k,i)\, R\, u_B(k,i), \quad (22)$$

where $\epsilon(k,i) = e_r(k+i) - e(k+i|k)$ is difference between reference (desired) trajectory tracking error $e_r(k+i)$ and predicted trajectory tracking error $e(k+i|k)$, h is prediction horizon interval, and Q and R are weighting matrices, where $Q \geq 0$, $Q \in \mathbb{R}^n \times \mathbb{R}^n$ and $R \geq 0$, $R \in \mathbb{R}^m \times \mathbb{R}^m$, n

is the number of state variables and m is the number of input variables.

For the design purpose, error dynamics (14) is discretized as

$$e(k+1) = Ae(k) + Bu_B(k), \tag{23}$$

where $A \in \mathbb{R}^n \times \mathbb{R}^n$ and $B \in \mathbb{R}^n \times \mathbb{R}^m$. Discrete model matrices A and B for short sampling time T_s can be well approximated with

$$\begin{aligned} A &= I + A_c T_s \\ B &= B_c T_s. \end{aligned} \tag{24}$$

Following [4], error prediction at the time instant h can be written as

$$\begin{aligned} e(k+h|k) = &\prod_{j=1}^{h-1} A(k+j|k)e(k) + \sum_{i=1}^{h} \left(\prod_{j=i}^{h-1} A(k+j|k) \right) \\ &\times B(k+i-1|k)u_B(k+i-1) \\ &+ B(k+h-1|k)u_B(k+h-1). \end{aligned} \tag{25}$$

Now the vector of trajectory tracking error predictions is defined as

$$E^*(k) = \left[e(k+1|k)^T \; e(k+2|k)^T \; \dots \; e(k+h|k)^T \right]^T \tag{26}$$

where $E^* \in \mathbb{R}^{n \cdot h}$. If the control vector is defined as

$$U_B(k) = \left[u_B(k)^T \; u_B(k+1)^T \; \dots \; u_B(k+h-1)^T \right]^T, \tag{27}$$

where $U_B \in \mathbb{R}^{m \cdot h}$, and

$$\Lambda(k,i) = \prod_{j=i}^{h-1} A(k+j|k), \tag{28}$$

the vector of trajectory tracking error predictions can be written in matrix form as

$$E^*(k) = F(k)e(k) + G(k)U_B(k), \tag{29}$$

where

$$F(k) = \left[A(k|k) \quad A(k+1|k)A(k|k) \quad \dots \quad \Lambda(k,0) \right]^T, \tag{30}$$

and

$$G(k) = \begin{bmatrix} B(k|k) & 0 & \cdots & 0 \\ A(k+1|k)B(k|k) & B(k+1|k) & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \Lambda(k,1)B(k|k) & \Lambda(k,2)B(k+1|k) & \cdots & B(k+h-1|k) \end{bmatrix}, \tag{31}$$

where $F(k) \in \mathbb{R}^{n \cdot h} \times \mathbb{R}^n$, $\quad G(k) \in \mathbb{R}^{n \cdot h} \times \mathbb{R}^{m \cdot h}$.

The reference trajectory tracking error can be defined as

$$e_r(k+1) = A_r^i e(k), \quad i = 1, \dots, h. \tag{32}$$

Matrix A_r is defined so that reference trajectory tracking error decreases with desired dynamics over time. Now the vector of reference trajectory tracking errors is defined as

$$E_r^*(k) = \left[e_r(k+1|k)^T \; e_r(k+2|k)^T \; \dots \; e_r(k+h|k)^T \right]^T, \tag{33}$$

and it can be computed as

$$E_r^*(k) = F_r e(k), \tag{34}$$

where

$$F_r = \begin{bmatrix} A_r & A_r^2 & \dots & A_r^h \end{bmatrix}^T, \tag{35}$$

and $F_r \in \mathbb{R}^{n \cdot h} \times \mathbb{R}^n$.

The cost function (22) can now be expressed as

$$J(U_B) = (E_r^* - E^*)^T \overline{Q} (E_r^* - E^*) + U_B^T \overline{R} U_B. \tag{36}$$

In order to obtain optimal control law, the cost function is minimized by setting its derivative to zero. In this way the control law becomes

$$U_B(k) = \left(G^T \overline{Q} G + \overline{R} \right)^{-1} G^T \overline{Q} (F_r - F) e(k), \tag{37}$$

where

$$\overline{Q} = \begin{bmatrix} Q & 0 & \cdots & 0 \\ 0 & Q & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & Q \end{bmatrix}, \overline{R} = \begin{bmatrix} R & 0 & \cdots & 0 \\ 0 & R & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & R \end{bmatrix}, \tag{38}$$

where $\overline{Q} \in \mathbb{R}^{n \cdot h} \times \mathbb{R}^{n \cdot h}$ and $\overline{R} \in \mathbb{R}^{m \cdot h} \times \mathbb{R}^{m \cdot h}$. The feedback control law of the MPC controller now becomes

$$u_B(k) = K_{mpc} \cdot e(k) \tag{39}$$

where K_{mpc} is defined as first m rows of the matrix $\left(G^T \overline{Q} G + \overline{R} \right)^{-1} G^T \overline{Q} (F_r - F)$, so that $K_{mpc} \in \mathbb{R}^m \times \mathbb{R}^n$.

IV. Experimental results

The developed algorithms were tested on robot soccer platform that is ideal for testing mobile robot related algorithms [7]. It consists of a team of five wheeled radio-controlled microrobots of size 7.5 cm cubed, with maximum velocity 2.4 m/s and maximum acceleration 2.5 m/s^2. The playground is of size 2.2×1.8 m. Above the center of the playground (2.5 m height), IEEE-1394 digital color camera with resolution of 656×494 pixels with maximal framerate of 80 fps was mounted (model Basler a301fc). Camera is connected to central computer, which executes vision algorithm, and also makes decisions and controls robots by transmitting referent linear and angular velocities to robots via wireless communication.

For experiments 8-shaped trajectory was used that is defined by

$$\begin{aligned} x_r(t) &= 1.1 + 0.7 sin\left(\frac{1.5t}{\sqrt{2.45}} \right) \\ y_r(t) &= 0.9 + 0.7 sin\left(\frac{2 \cdot 1.5t}{\sqrt{2.45}} \right). \end{aligned} \tag{40}$$

This type of trajectory is appropriate for experiments, because it has sharp turns, as well as parts with high acceleration and velocity, but it's still feasible for the robot because there are no step changes of the tangential, nor angular velocity, and maximum acceleration of the robot is never exceeded.

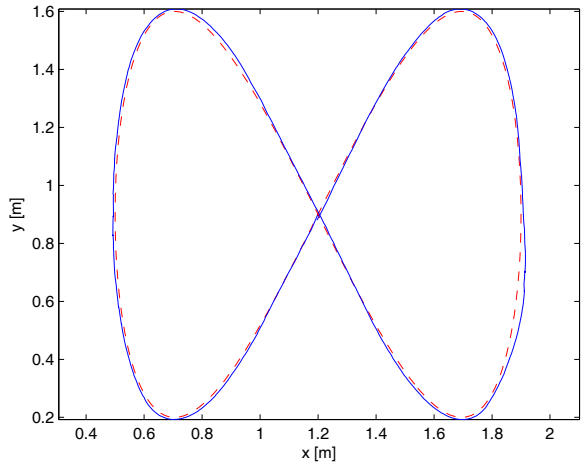

Fig. 3. Trajectory tracking experiment with linear controller: robot path (–), reference path (- -)

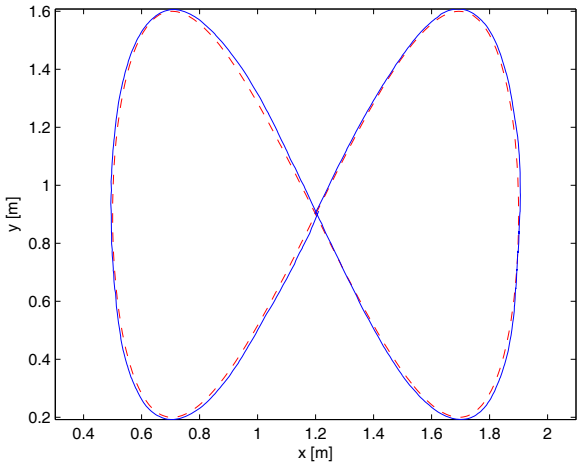

Fig. 4. Trajectory tracking experiment with nonlinear controller: robot path (–), reference path (- -)

The maximum tangential velocity of this trajectory is 1.5 m/s, and maximum tangential acceleration is 1.9 m/s^2, which is close to robot limits. In experiments robot was first accelerated to achieve position, orientation and velocity that are approximately equal to trajectory initial condition.

The sampling period for all controllers was 12.5 ms. The parameters for linear controller were $\zeta = 0.7$ and $g = 60$, and the same parameters were used for nonlinear controller. For the MPC controller, after many experiments the prediction horizon $h = 12$ was chosen as best, the reference matrix was $A_r = I_{3\times3} \cdot 0.85$, and weighting matrices were

$$Q = \begin{bmatrix} 4 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 0.1 \end{bmatrix}, \quad R = I_{2\times2} \times 10^{-3}.$$

The obtained robot paths for experiments with linear, nonlinear and MPC controller are given in Figures 3, 4 and 5, respectively. The correspondent sum of squared errors (SSE) for each component of error vector e are

$$\begin{aligned} \text{SSE}_{linear} &= (0.0177 \quad 0.0397 \quad 0.4395) \\ \text{SSE}_{nonlinear} &= (0.0144 \quad 0.0354 \quad 0.3555) \\ \text{SSE}_{MPC} &= (0.0442 \quad 0.0416 \quad 0.4010). \end{aligned}$$

It can be seen that nonlinear controller has the lowest SSE for all three error components, although this difference is not significant which can also be confirmed visually by comparing obtained paths. The error of linear controller is only slightly higher compared to nonlinear. But although in simulations MPC gives lower tracking error than linear and nonlinear controllers, in real world it is no more the case as it results with somewhat higher longitudinal error e_1 while other two error components are comparable with other controllers.

The produced command robot velocities of linear, nonlinear and MPC controller experiments, together with ideal commands obtained from reference trajectory, are given in

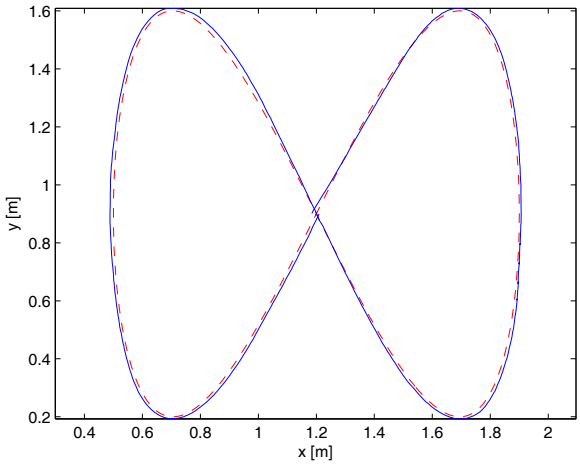

Fig. 5. Trajectory tracking experiment with MPC controller: robot path (–), reference path (- -)

Figures 6, 7 and 8, respectively It can be observed that all controllers produce similar command values but MPC controller's command values u_1 and u_2 are closest to ideal command tangential v_r and angular ω_r velocities obtained from reference trajectory.

Regarding the parameters choice, for linear and nonlinear controllers it was very easy to choose controller parameters, but MPC controller was very sensitive to parameters choice and control easy became unstable with wrong choice. Regarding the robustness, at high velocities the linear and nonlinear controller showed acceptable robustness in presence of measurement noise and small initial robot position error, as opposed to MPC controller which often showed unstable behavior in presence of noise and initial error. This was also proved by the fact that the maximum velocity that could be obtained was about 1.5 m/s for MPC, 1.6 m/s for linear and

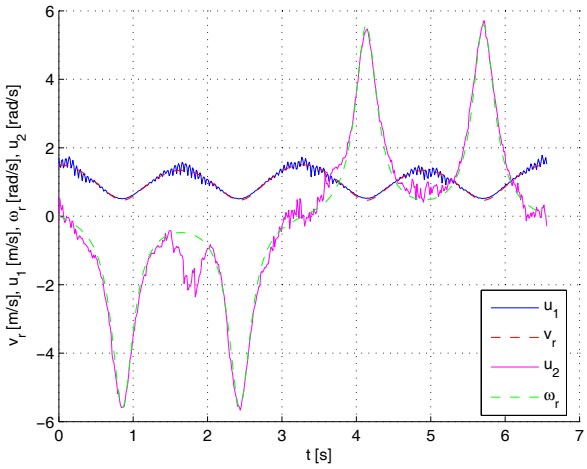

Fig. 6. Experiment with linear controller: reference tangential (v_r) and angular (ω_r) velocity, command tangential (u_1) and angular (u_2) velocity

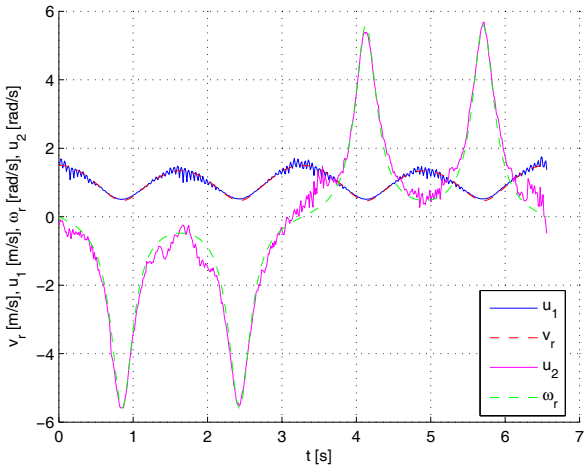

Fig. 7. Experiment with nonlinear controller: reference tangential (v_r) and angular (ω_r) velocity, command tangential (u_1) and angular (u_2) velocity

1.7 m/s for nonlinear controller.

V. Conclusion

An experimental comparison of three trajectory tracking controllers for nonholonomic mobile robot is presented: linear controller, nonlinear controller and model predictive controller. The goal was to obtain highest possible trajectory velocity, and all controllers were tested on a real mobile robot. Experiments have shown that all three controllers produce comparable quality of trajectory tracking, but MPC controller has high sensitivity to initial robot state error, and it is very difficult to determine optimal parameters of the MPC algorithm. Moreover, the drawback of the MPC controller is its complexity, especially if longer prediction intervals are used. Therefore, at high velocities the nonlinear controller is recommended as it provides global asymptotic stability, it is not difficult to choose its parameters and is simple to implement and has low computational requirements which make it especially appropriate for fast, real-time trajectory tracking.

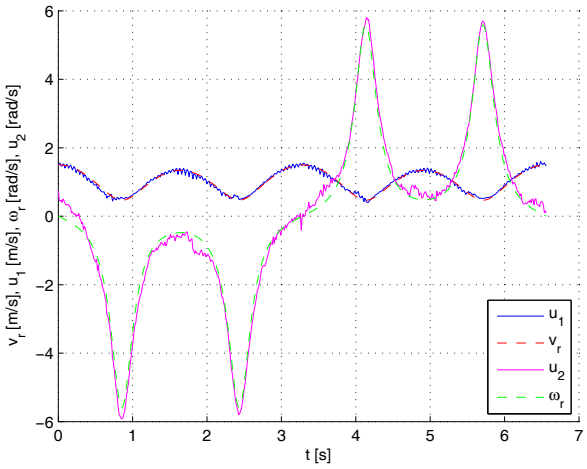

Fig. 8. Experiment with MPC controller: reference tangential (v_r) and angular (ω_r) velocity, command tangential (u_1) and angular (u_2) velocity

Future work will include developing and comparison of more advanced controller schemes, like dynamic feedback linearization based controller and nonlinear model predictive controller. Also, the procedure of determining optimal controller parameters will be developed.

Acknowledgment

This research was supported by the Ministry of Science, Education and Sports of the Republic of Croatia (grant No. 036-0363078-3018).

References

[1] A. D. Luca, G. Oriolo, and M. Vendittelli, *Control of wheeled mobile robots: An experimental overview, in:S. Nicosia, B. Siciliano. A. Bicchi, p: Valigi, (Eds.) RAMSETE - Articulated and Mobile Robotics for Services and Technologies*, 3rd ed. Springer-Verlag, 2001.

[2] A. Ollero and O. Amidi, "Predictive path tracking of mobile robots," in *Proceedings of 5th International Conference on Advanced Robotics, Robots in Unstructured Environments (ICAR '91)*, vol. 2, June 1991, pp. 1081 – 1086.

[3] J. E. Normey-Rico, J. Gmez-Ortega, and E. F. Camacho, "A smith-predictor-based generalised predictive controller for mobile robot path-tracking," *Control Engineering Practice*, vol. 7, no. 6, pp. 729 – 740, 1999.

[4] G. Klancar and I. Skrjanc, "Tracking-error model-based predictive control for mobile robots in real time," *Robotics and Autonomous Systems*, vol. 55, pp. 460–469, 2007.

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

[6] C. Samson, "Time-varying feedback stabilization of car-like wheeled mobile robots," *International Journal of Robotics Research*, vol. 12, no. 1, pp. 55 – 64, 1993.

[7] H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, and E. Osawa, "RoboCup: The robot world cup initiative," in *Proceedings of the First International Conference on Autonomous Agents (Agents'97)*. New York: ACM Press, 1997, pp. 340–347.