International Journal of Advanced Robotic Systems

OPEN ACCESS
ARTICLE

Testing PID and MPC Performance for Mobile Robot Local Path-following

Regular Paper

Lluis Pacheco[1]* and Ningsu Luo[1]

1 University of Girona, Girona, Spain
*Corresponding author(s) E-mail: lluispa@eia.udg.edu

Received 21 July 2014; Accepted 31 July 2015

DOI: 10.5772/61312

Abstract

This paper outlines the online performance of a control law based on PID (proportional-integral-derivative) controllers and MPC (model predictive control) for mobile robot local path-following. Both techniques share the use of a set of different dynamic models. PID controllers are used for controlling the speed of the robot's wheels, while high level algorithms compute the necessary wheel speeds in order to generate a motion that approaches the vehicle towards the desired path. Meanwhile, local MPC is implemented by computing the horizon of suitable coordinates that arise from the set of command input combinations. Therefore, command speeds that correspond to the desired point are obtained by minimizing a cost function in which the population of the available coordinates is taken into account.

Keywords Mobile Robots, Local Path-following, Model-based PID and Predictive Control, Experimental Factor Tuning

1. Introduction

PID (proportional-integral-derivative) controllers are the most dominating form of feedback that is in use today. It is quite reasonable to predict that they will continue to be used in the future [1]. MPC (model predictive control) is a very effective advanced control technique, compared to standard PID controllers, and has had a significant impact on industrial process control [2-3]. Recently, online LMPC (local model predictive control) has been suggested as a suitable solution for the motion control of mobile robots [4]. This paper outlines the performance of PID and LMPC techniques that are applied to a differentially driven WMR (wheeled mobile robot). Its objective is to analyse the statistical results obtained from real-time experiments, with the two control systems being presented in a single study within a unified framework.

Differential driven WMRs are considered to be nonholonomic systems [5]. From a control point of view, the accuracy and performance of path-following algorithms are subject to nonholonomic constraints. Path-following can be achieved using time-varying, discontinuous or hybrid feedback laws [6]. Sørdalen and Canudas de Wit reports the use of piecewise analytic feedback laws in a discontinuous stabilization approach to an arbitrary point. This approach is used to track a sequence of fixed points that consist of positions and orientations. The desired path can be composed of a sequence of straight lines and circle segments [7]. Thus, instead of regulating the robot about one point, the sequence of points to track is determined by moving the WMR from point-to-point in the state space, using feedback combined with some path planning [8]. A path-following motion task can be determined by giving a

sequence of waypoints, which have to be passed by the WMR [9].

Linear control is a suitable methodology when control law discontinuities, which refer to path-following, are considered. Within this scope, PID solutions are widely adopted as low level DC (direct current) motor speed controllers [10]. Path-following achievement is accomplished by developing high level strategies for motion planning. For instance, the dynamic window approach is based on the dynamic constraints of a WMR. It uses the available robot speeds to plan avoidance of collisions with obstacles, safety stops and achievement of goals [11]. Sliding mode controllers with discontinuous and smooth actuations are also proposed by Matveev, Hoy, Katupitiya and Savkin [12]. In the controller design, they consider a set of constraints on the steering angle.

Other suitable control approaches have proposed the development of a neural controller as a function of the sensor inputs in order to change a WMR's behaviour so that it can perform different tasks [13]. Capi points out the importance of using real robot systems for achieving effective learning by neural controllers. In Capi's studies, empirical results are used as an important data source to achieve proper WMR navigation. Cheng reports successful path-following by mobile robots using fuzzy control systems [14]. The use of adaptive control capabilities that include robot-model uncertainties are also proposed in Pourboughrat and Karlsson [15].

Regarding MPC, some approaches that use mobile robots have proposed the use of potential fields, which satisfy the stability criteria in a Lyapunov sense. Convergence to goals and reactive behaviour can be achieved by formulating the two objectives within an MPC and CLF framework [16]. Most of the research developed using MPC techniques and their application to WMRs is based on the assumption that the reference trajectory is known beforehand. The use of mobile robot kinematics to predict future system outputs has been proposed in most of the approaches described in previous literature [17-18]. The use of kinematics should include velocity and acceleration constraints to prevent infeasible path-following objectives for the WMR [17]. MPC techniques that are based on a linear time-varying description of system dynamics have also been proposed [19]. Kühne describes the possibility of using linear time-varying descriptions of the system, as well as real-time implementations, with short prediction horizons. Maximizing the use of data and a priori knowledge, and minimizing the impact and effort brought about by process and regulatory control changes, has also been suggested [20]. In this study, dynamic input-output models have been proposed as a way to include dynamic constraints within the system model for controller design. The use of a short prediction horizon makes it possible to implement obstacle-avoidance policies and final-goal achievement approaches. Moreover, control effort is constrained.

The aim of this work is to make up for the lack of published research in relation to LMPC and PID analyses and experimental results when both controllers are considered together. This paper outlines the performance of PID and LMPC techniques applied to a differentially driven WMR. The main contribution is to present the testing results of both of the low level controllers when local path-following is considered, in which online LMPC is a suitable controller when the precise motion control is pursued.

The research introduces online LMPC and control laws that are based on PID controllers, which use dynamic models for local path-following. Knowledge of different models can provide information about the dynamics of a robot and, consequently, about its reactive parameters and turning capabilities, as well as safety-stop distances [21]. In the cases studied in this paper, three different dynamic models are proposed. Low level PID control systems are used as speed controllers in the majority of the research that has been carried out. Motion control for path-following is implemented, employing discontinuous control laws that use heuristic concepts. Developed research into PID and LMPC controllers uses the heading-error and the path-deviation of the robot as the necessary motion control parameters. Deviation errors are used in the control law as information that is necessary for going forward or turning. This research focuses on experimental results by implementing factorial design experiences [22]. The performance of PID and LMPC controllers is studied by commanding a sequence of waypoints that define the path, and analysing statistics of parameters such as time, path-deviation and velocities. This paper is organized as follows. Section 2 introduces a description of WMR kinematic and dynamic systems and depicts the experimental methodology used to obtain the various dynamic models. In Section 3, PID controllers are obtained and their performance is experimentally tested. High level strategies for motion planning are also introduced. Section 4 depicts the MPC formulation for path-following, as well as the preliminary experimental results, which have been developed for adjusting the control law parameters. In Section 5, PID and LMPC are tested by using the same paths. From the statistical results obtained, the performance of both methods is analysed. Conclusions and descriptions of future work are given in Section 6.

2. Mobile Robot Kinematic and Dynamic Systems

This section presents the WMR kinematic models and the dynamic models obtained from the experimental results. A differentially driven WMR with a free rotating wheel is used for the experimental study. The WMR structure is made of aluminium, see Figure 1. It consists of different levels where the constitutive parts are placed. The two differential driven wheels, which are controlled by two dc motors, and the third omni-directional wheel, which gives the third contact point with the floor, are placed on the first

level. The embedded PC computer is placed on the second level. The sonar sensors and specific hardware are placed on the third level. The machine vision system is placed on the fourth level. The hardware and robot architecture descriptions can be found in [4]. Figure 1 shows the available WMR platform, designed for indoor navigation. In the following subsections, the robot's kinematic and dynamic systems are described. Particular attention is paid to the experiments developed for obtaining the set of dynamic models that are used in this work.

2.1 Kinematic System

The WMR is a rigid body and consequently, non-deforming wheels are required. The robot is assumed to move without slipping on a plane, with pure rolling contact between the wheels and the ground. Denoting the position and orientation coordinates by (x, y, θ) and the velocity vector by $u = [v, w]^T$, where v and w denote the tangential and angular velocities respectively, the kinematic model of the WMR can be stated as:

$$\begin{aligned} \frac{dx}{dt} &= v\cos\theta \\ \frac{dy}{dt} &= v\sin\theta \\ \frac{d\theta}{dt} &= \omega \end{aligned} \tag{1}$$

Using a discrete-time representation (with T being the sampling period and k being the time instant) and Euler's approximation, the following discrete-time model can be obtained for the robot's dynamics:

$$\begin{aligned} x(k+1) &= x(k) + v(k)\cos\left(\theta(k) + \omega(k)T\right) \\ y(k+1) &= y(k) + v(k)\sin\left(\theta(k) + \omega(k)T\right) \\ \theta(k+1) &= \theta(k) + \omega(k)T \end{aligned} \tag{2}$$

The WMR platform uses incremental encoders to obtain the position and orientation coordinates. The positioning of the robot is a function of the radius of the left and right wheels (R_l, R_r) and angular incremental positioning (θ_l,θ_r), with E being the distance between the two wheels and dS being the incremental displacement of the robot. The position and angular incremental displacements can be expressed as:

$$\begin{aligned} dS &= \frac{R_r d\theta_r + R_l d\theta_l}{2} \\ d\theta &= \frac{R_r d\theta_r - R_l d\theta_l}{E} \end{aligned} \tag{3}$$

The (x, y, θ) coordinates can be expressed as:

$$\begin{aligned} x(k+1) &= x(k) + dS\cos\left(\theta(k) + d\theta\right) \\ y(k+1) &= y(k) + dS\sin\left(\theta(k) + d\theta\right) \\ \theta(k+1) &= \theta(k) + d\theta \end{aligned} \tag{4}$$

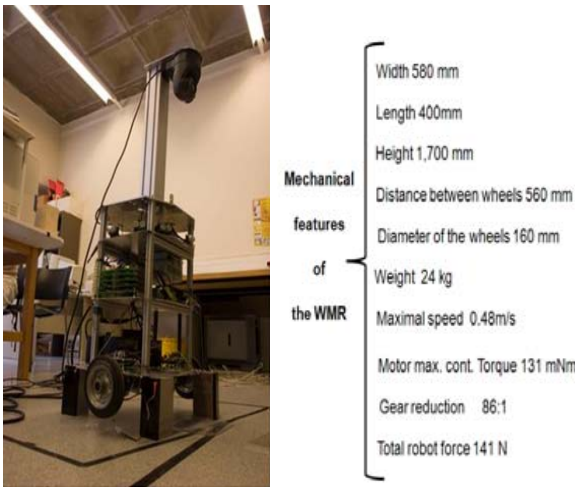

Figure 1. WMR PRIM

Thus, the incremental position of the robot can be obtained using the odometer system and the encoder information available from equations (3) and (4).

2.2 Experimental Dynamic Models

This subsection deals with the problem of modelling the dynamics of a WMR system. The aim is to obtain a set of dynamic models for high, medium and low velocities that are suitable for use in controlling the velocity of each wheel. These models use PID controllers and predict the range of possible coordinates when MPC strategies are used. Dynamics constraints, such as the range of speed and acceleration, are included in the dynamic models. The model is obtained using a set of linear transfer functions, which represent the nonlinearities of the whole system. The parameter identification process is based on black-box models [23-24]. The nonholonomic system that is dealt with in this work is considered initially as a MIMO (multiple-input multiple-output) system. This is because of the dynamic influence between the two DC motors. This MIMO system is composed of a set of SISO (single-input single-output) subsystems with coupled connections. The parameter estimation is performed using a PRBS (pseudo-random binary signal) as an excitation input signal. It guarantees the correct excitation of all of the dynamically sensitive modes of the system across the whole spectral range, resulting in an accurate high-precision parameter estimation. The experiments to be performed consist of exciting the two DC motors in various (low, medium and high) speed ranges. An ARX (autoregressive with external input) structure is used to identify the system parameters. The

problem lies in finding a model that minimizes the error between the real and estimated data. By expressing the ARX equation as a linear regression, the estimated output can be written as:

$$\hat{y} = \lambda\phi \tag{5}$$

where $\hat{y}$ is the estimated output vector, λ the vector of the estimated parameters, and φ the vector of the measured input variables. Using a coupled system structure, the transfer function of the robot can be expressed as follows:

$$\begin{pmatrix} Y_R \\ Y_L \end{pmatrix} = \begin{pmatrix} G_{RR} & G_{LR} \\ G_{RL} & G_{LL} \end{pmatrix} \begin{pmatrix} U_R \\ U_L \end{pmatrix} \tag{6}$$

where Y_R and Y_L represent the speeds of the right and left wheels respectively and U_R and U_L the corresponding speed commands. To obtain information about the coupled system, the matrix of the transfer function (G_{RR}, G_{LR}, G_{RL}, G_{LL}) needs to be identified. The filtered data, which represent the average values of five different experiments with the same input signal, are used for identification. The system is identified using MATLAB's "ident" identification toolbox for second-order models. Table 1 shows the continuous transfer functions obtained for the three different linear speed models. The coupling effects show a way of obtaining a reduced-order dynamic model. It can be seen from Table 1 that the dynamics of the two DC motors are different and the steady-state gains of the coupling terms are relatively small (less than 20% of the gains of the main diagonal terms). Therefore, it is reasonable to ignore the coupling dynamics in order to obtain a simplified model. In order to verify this using real results, a set of experiments was performed, sending a zero speed command to one motor and various non-zero speed commands to the other. The result confirmed that the coupled dynamics could be ignored. The existence of different steady-state gains was also verified experimentally. Finally, the order reduction of the system model was carried out through an analysis of the pole positions using the root locus method. This approach revealed the existence of a dominant pole. Consequently, the model order could be reduced from second-order to first-order. Table 2 shows the first-order transfer functions obtained. Afterwards, the system models could be validated with experimental data using the PRBS input signal.

The dynamic behaviour of the WMR is represented by the above linear models, obtained experimentally via the identification process. In this way, the set of outputs obtained using the dynamics model can reproduce the kinematics and constraints of the real system.

Linear Transfer Function	High Velocities (0.45m/s)	Medium Velocities (0.3m/s)	Low Velocities (0.2m/s)
G_{RR}	$\frac{0.2s^2 + 3.2s + 9.4}{s^2 + 6.6s + 9.9}$	$\frac{0.2s^2 + 3.1s + 8.4}{s^2 + 6.2s + 9.1}$	$\frac{0.2s^2 + 2.3s + 5.4}{s^2 + 5.2s + 6.6}$
G_{LR}	$\frac{-0.04s^2 - 0.6s - 0.3}{s^2 + 6.6s + 9.9}$	$\frac{-0.02s^2 - 0.3s - 0.03}{s^2 + 6.2s + 9.1}$	$\frac{-0.02s^2 - 0.2s + 0.4}{s^2 + 5.2s + 6.6}$
G_{RL}	$\frac{-0.01s^2 + 0.1s - 0.4}{s^2 + 6.6s + 9.9}$	$\frac{-0.01s^2 + 0.1s - 0.4}{s^2 + 6.2s + 9.1}$	$\frac{-0.01s^2 - 0.1s - 0.2}{s^2 + 5.2s + 6.6}$
G_{LL}	$\frac{0.3s^2 + 4.5s + 9}{s^2 + 6.56s + 9.9}$	$\frac{0.3s^2 + 4.5s + 8.97}{s^2 + 6.2s + 9.1}$	$\frac{0.3s^2 + 3.5s + 6.3}{s^2 + 5.2s + 6.6}$

Table 1. The second order model

Linear Transfer Function	High Velocities (0.45m/s)	Medium Velocities (0.3m/s)	Low Velocities (0.2m/s)
G_{RR}	$\frac{0.95}{0.42s + 1}$	$\frac{0.92}{0.41s + 1}$	$\frac{0.82}{0.46s + 1}$
G_{LL}	$\frac{0.91}{0.24s + 1}$	$\frac{0.92}{0.27s + 1}$	$\frac{0.95}{0.33s + 1}$

Table 2. The reduced WMR model

3. Design of PID Controllers and Motion Planning

This section introduces the design of low level PID speed controllers. High level motion planning is also described. Experimental control law factor tuning is analysed by considering the kind of path to be followed.

3.1 Design of PID Controllers

PID control design is a well-known problem that is solved by considering a closed loop system, which includes the plant and the controller [10]. The problem is first tackled in a continuous time domain, employing the first order models shown in Table 2 and PI controllers. It is outlined that the results presented in this paper are based on the PI controllers depicted in Table 3.

A second order transfer function, where the PI parameters can be adjusted, is obtained by selecting an appropriate damping ratio, and using other design parameters such as overshoot ratio or settling time. In this study, the damping ratio δ is set to 0.7, the overshoot ratio to 10% and the settling time to 2s. After computing continuous-time PI controller parameters, the appropriate computer control factors are obtained by continuous to discrete time transforms. The step response of the system is used to compute an adequate sampling time [25]. In this study, the sampling time is set to 100 ms. Table 3 shows the continuous and discrete controller transfer functions obtained between U (speed commands assigned to each DC motor) and e (the difference between the velocities commanded and obtained), where "s" is the Laplace variable and "z" is the discrete-time variable.

The performance of PI controllers is experimentally verified. Figure 2 shows, respectively, the speed, error and first derivative of the error for low velocities.

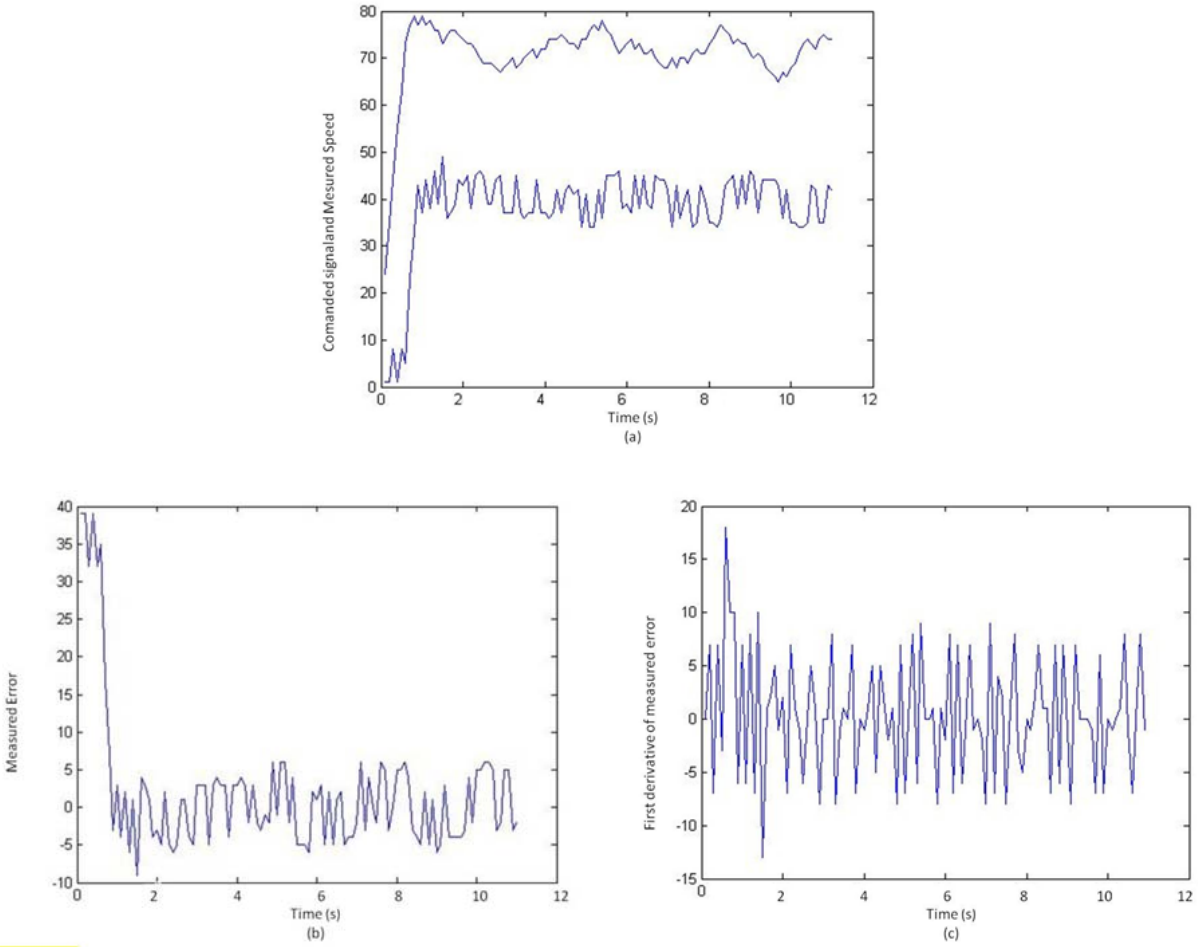

Figure 2. (a) Command signal and measured speed of left wheel for low velocities. (b) Speed error. (c) First derivative of the speed error.

It is noted that the commanded speed and the measured speed are represented by binary values from 0 to 128. In this way, the speed value of 40 corresponds to the velocity of 0.25m/s.

Similar results were obtained for the complete set of controllers, as shown in Table 3.

Continuous PI Controller Transfer Functions			
	High Velocity	**Medium Velocity**	**Low Velocity**
Right Wheel	$\frac{0.59s + 3.36}{s}$	$\frac{0.70s + 3.65}{s}$	$\frac{1.02s + 4.59}{s}$
Left Wheel	$\frac{0.04s + 2.16}{s}$	$\frac{0.33s + 2.40}{s}$	$\frac{0.33s + 2.81}{s}$
Discrete PI Controller Transfer Functions			
	High Velocities	**Medium Velocities**	**Low Velocities**
Right Wheel	$\frac{0.59 - 0.25z^{-1}}{1 - z^{-1}}$	$\frac{0.7 - 0.54z^{-1}}{1 - z^{-1}}$	$\frac{1 - 0.54z^{-1}}{1 - z^{-1}}$
Left Wheel	$\frac{-0.04 + 0.54z^{-1}}{1 - z^{-1}}$	$\frac{0.33 - 0.09z^{-1}}{1 - z^{-1}}$	$\frac{0.33 - 0.05z^{-1}}{1 - z^{-1}}$

Table 3. Continuous and discrete PI controllers

3.2 Motion Planning

Path-following achievement is accomplished by tracking a sequence of fixed points consisting of positions and orientations. Feedback of heading orientations and Cartesian coordinates is done for reducing the path-deviation.

The motion for following the desired path is composed by a sequence of going straight and turning actions. Figure 3 shows a path defined by three waypoints. It is noted that between two consecutive waypoints, the path is given by the straight line that links the two points. To perform a straight-path-following action, both wheels must have the same velocity, which means keeping the heading angle constant.

Right and left turning actions are performed by the following algorithms:

$$\begin{aligned} U_R(k+1) &= U_R(k) + K_R\left(\theta_d - \theta(k)\right) \pm \Delta U_{PD} \\ U_L(k+1) &= U_L(k) + K_L\left(\theta(k) - \theta_d\right) \pm \Delta U_{PD} \end{aligned} \tag{7}$$

U_R and U_L denote the commanded speed for the right and left wheels respectively. θ_d depicts the desired orientation

and θ the WMR's heading orientation, see Figure 3. K_R and K_L are tuning factors for right and left turning of the wheels.

This means that when a straight path is followed, the difference between θ_d and θ is zero, and straight path-following is accomplished by commanding the same velocity to each wheel. When left turning has to be done, θ_d is larger than θ and the computation in equation (7) increases U_R, while U_L is decreased. Conversely, when right turning has to be accomplished, θ is larger than θ_d and the computation in (7) increases U_L, while U_R is decreased. ΔU_{PD} is a turning parameter that reduces the path-deviation when Cartesian coordinates are considered. It is applied with a different sign to both wheels with the aim of creating a difference of velocities.

Path-deviation is given by the Euclidean distance between the WMR coordinates, (Rx, Ry), and the point, (Px, Py). The point (Px, Py) is defined by the intersection of the line that passes through (Rx, Ry) and it is perpendicular to the straight line, which is defined by the consecutive waypoints that define the path that is followed, see Figure 3.

ΔU_{PD} is considered as a heuristic parameter with three different values:

- ΔU_{PD} is equal to zero when the path-deviation is very small.
- ΔU_{PD} acts as a factor that produces a slight turn when the path deviation is small.
- ΔU_{PD} acts as a factor that produces a turn when the path deviation is important.

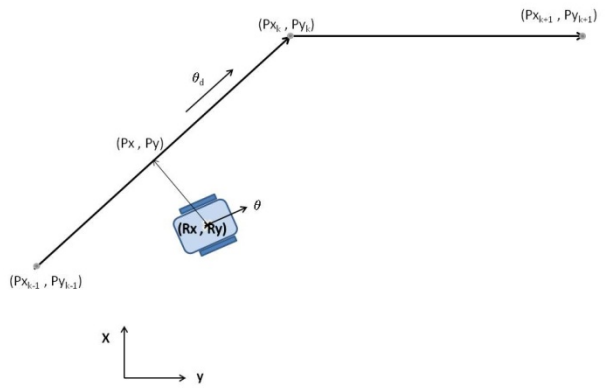

Figure 3. Example of a path given by the waypoints (Px_{k-1}, Py_{k-1}), (Px_k, Py_k) and (Px_{k+1}, Py_{k+1})

Euclidean path-deviations are reduced by using turning actions that approach the vehicle to the path. In this way, turning actions are a function that depends on the segment being followed and the relative WMR position. Figure 3 depicts a WMR pose, where the X coordinate Rx has a value that is less than the X coordinate of the distance to the path Px. Path-deviation is reduced by increasing the right wheel speed and reducing the left wheel speed. Regarding relative robot position and the orientation of the different path-following segments, we consider the following situations:

- The desired heading for each segment is given by the slope, which is defined by the two extreme points that define the segment being followed. This is classified as belonging to one of the four quadrants.
- Each quadrant depicts a different action for reducing the path-deviation.
- The relative position, up or down, of the WMR and the path segment produces opposite turning actions.
- Horizontal and vertical path-following segments are considered as special cases. Horizontal orientation has a fixed X coordinate. Zero or π orientation case is selected by comparing the values of the Y Euclidean coordinates that define the segment being followed. Vertical orientation has a fixed Y Cartesian coordinate. The $\pi/2$ or $(5\pi)/2$ orientation case is selected by comparing the values of the X Cartesian coordinates that define the segment being followed.

It is outlined that terms that are small or important deviations are experimentally adjusted as a function of the WMR features. Moreover, a high level motion planner takes into consideration other factors:

- Acceleration ramps have been implemented.
- The speed of the WMR (slow, medium or high) is a function of the distance to the point to be tracked.
- When the path-followed distance approaches the desired point, the velocities are reduced. So, if no more points are commanded, the WMR has to stop before the last commanded point.
- Saturation of wheel velocities is avoided.

The proposed algorithms have been tested by considering five different kinds of necessary motions used for performing local path-following (straight, wide left turn, slight left turn, wide right turn and slight right turn). We consider a path as straight when the orientation error is less than 5°. A slight turn should be commanded when the left or right heading error is equal to or larger than 5° and less than 25°. A wide turn should be commanded when the orientation error is larger than or equal to 25°. The on-robot computational cost is less than 1ms and factor tuning of K_R and K_L parameters is done by considering experimental data [22]. K_R and K_L are the parameters that relate to the heading angle measurement and have to be sensitive to heading errors. Using degrees as the unit of measurement, initial values close to one are proposed in order to perform their factor tuning. High and low values are set by slightly modifying the values that correspond to sensitive parameters and verifying their validity for a set of paths. Heuristic experiments for selecting high and low values of K_R and K_L have depicted a suitable narrow interval of around 10%. The

ΔU_{PD} parameter is set to zero when the path-deviation is less than 5 cm. It is set to 5 command units of speed when the path-deviation is between 5 and 20 cm, and to 10 command units of speed when the path-deviation is larger than 20 cm. It is noted that ΔU_{PD} is a value that is added to the speed command value, which is a value from 0 to 127, with a dead zone from 0 to 20 and where an output speed of 90 is equivalent to 0.5 m/s.

4. Online LMPC Techniques for Path-following

In this section, online LMPC techniques that are based on the dynamic models obtained in Section 2 are introduced. The use of dynamic models avoids the need for the velocity and acceleration constraints that are used in other MPC studies that are based on kinematic models. Real-time implementations are easy to put into effect because short prediction horizons are used. With LMPC, the idea of a receding horizon can be used to deal with local on-robot sensor information. LMPC and contractive-constraint formulations, as well as the algorithms and control law tuning used, are introduced in the following subsections.

4.1 LMPC Formulation and Algorithms

The main objective of highly precise motion control is to minimize the error between the robot and the desired path. Global path planning becomes infeasible when the sensor system of a robot is entirely local [26]. LMPC is proposed as a way to use available local perception data in navigation strategies [4]. In concrete terms, LMPC is based on minimizing a cost function related to the objectives for generating the optimal WMR inputs. The cost function can be defined as follows:

$$J(n,m)=\min_{\left\{U(k+i|k)_{i=0}^{i=m-1}\right\}}\left\{\begin{array}{l}\left[X(k+n|k)-X_{ld}\right]^T P\left[X(k+n|k)-X_{ld}\right]\\ +\sum_{i=1}^{n-1}\left[\theta(k+i|k)-\theta_{ld}\right]^T Q\left[\theta(k+i|k)-\theta_{ld}\right]\\ +\sum_{i=1}^{n-1}\left[X(k+i|k)-\overline{X_{ld}X_{l0}}\right]^T R\left[X(k+i|k)-\overline{X_{ld}X_{l0}}\right]\\ +\sum_{i=0}^{m-1}U^T(k+i|k)RU(k+i|k)\end{array}\right\} \quad (8)$$

The first term of (8) refers to the attainment of the desired local coordinates, $X_{ld}=(x_d, y_d)$, where (x_d, y_d) denote the desired Cartesian coordinates, and $X(k+n/k)$ represents the terminal value of the predicted output after the horizon of prediction n. The second term is the predicted orientation error. The desired orientation is denoted by θ_{ld}, which represents the orientation pursued towards the local objective. The third term of (8) relates to the distance between the predicted robot positions and the path segment given by a straight line between the Cartesian coordinates $X_{l0}=(x_{l0}, y_{l0})$ of the robot when the desired coordinates are modified. The desired local positions, such as X_{ld}. $X(k+i/k)$ and $\theta(k+i/k)$ $(i=1,...n-1)$, represent the predicted Cartesian and orientation values within the prediction horizon. The last term relates to the power signals that are assigned to each DC motor, which are denoted as U. The parameters P, Q, R, and S are weighting parameters that express the importance of each term. The control horizon is designated by the parameter m. System constraints are also considered:

$$\left\{\begin{array}{c} G_0<\left|U(k)\right|\le G_1\\ \left|X(k+n/k)-X_{ld}\right|\le\alpha\left|X(k)-X_{ld}\right|\\ or\ \ \left|\theta(k+n/k)-\theta_{ld}\right|\le\alpha\left|\theta(k)-\theta_{ld}\right|\quad \alpha\in(0,1)\end{array}\right\} \quad (9)$$

where $X(k)$ and $\theta(k)$ denote the current WMR coordinates and orientation and $X(k+n/k)$ and $\theta(k+n/k)$ denote the final predicted coordinates and orientation, respectively. The limitation of the input signal is taken into account in the first constraint, where G_0 and G_1 denote the dead zone and saturation, respectively of the DC motors. The second and third terms are contractive constraints [27], which result in the convergence of coordinates or orientation to the objective. These should be evaluated at each control step. An LMPC algorithm consists of the following steps: 1. Read the current position; 2. Minimize the cost function and obtain a series of optimal input signals; 3. Choose the first input signal obtained as the command signal; 4. Go back to step 1 in the next sampling period.

It is noted that the optimal input is obtained using equation (8), which uses parameters related to an approach where the path is defined by local waypoints, see Figure 3. Therefore, it is not guaranteed that the optimization of equation (8) is optimal to path errors when a global path is analysed. The minimization of the cost function is dealt with by computing the complete input set as a way of obtaining the optimal input. To reduce the set of possibilities when an optimal solution is sought, certain constraints on the DC motor inputs are assumed:

- The signal increment is kept fixed within the prediction horizon.
- The input signals remain constant during the remaining time interval.

These assumptions will reduce computation time and ensure the smooth behaviour of the robot within the prediction horizon [2]. Thus, the set of available inputs is reduced to a single value, as shown in Figure 4. The set of available inputs can be represented by a bi-dimensional array, where different search algorithms can be tested. In this context, we considered other algorithms such as the gradient descent algorithm that divides the array in different regions. However, we obtained sub-optimal solutions in the presence of some local minimum failures. In this work, as consequence of the results obtained, we implement the complete input search algorithm.

The sampling time for each LMPC step is set to 100 ms, as it was for the PID controllers. The available coordinate predictions are computed by using a complete input search

for a short prediction horizon (*n=5, m=3*). The on-robot algorithm time performance is set to 45ms using an embedded 700-Mhz PC and additional on-robot hardware systems. Figure 5 depicts PID and LMPC computational times. The PID processing time is very short when compared with LMPC algorithms, which involve a complete input search. Surprisingly, when the LMPC horizon is increased, the computing time decreases. The reason for this can be found in the fact that the signal increment is kept fixed within the prediction horizon.

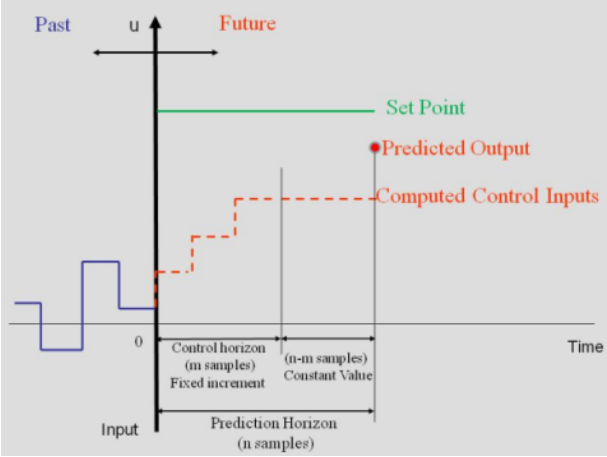

Figure 4. LMPC strategy with fixed increment of the input during the control horizon and constant value for the remaining time

Thus, the maximum input value possibilities decrease with larger horizons. For *n=5*, there are 1764 possibilities (*42x42*) and for n=10, there are 625 (*25x25*). The population of avalaible coordinates is denser when m=3 and n=5.

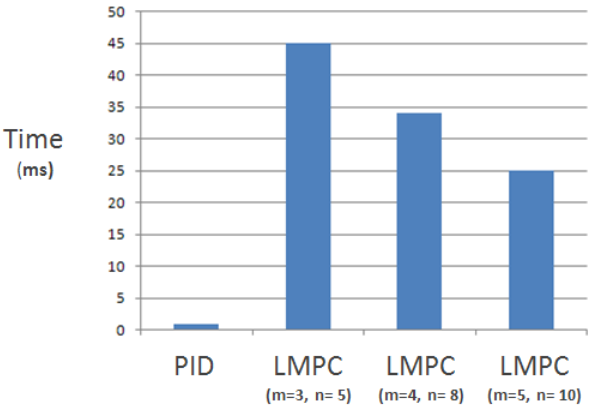

Figure 5. PID and LMPC processing time

It is noted that Figure 5 just shows the processing time of the low level PID controllers without considering the processing time of the high level algorithms.

4.2 Path-following Experimental Cost-function Tuning

Path-following performance can be improved by the choice of an appropriate cost function, derived from (8). This consists of a quadratic expression that contains some of the following four parameters to be minimized:

- The first parameter is the squared Euclidean *approaching-point distance* (APD) between the local desired coordinates and the actual robot position. This distance corresponds to parameter "*P*" of the LMPC cost function given by (8).
- The second parameter is the squared *orientation deviation* (OD), which considers the heading difference towards the desired orientation. The OD corresponds to parameter "*Q*" of the LMPC cost function in (8).
- The third parameter is the squared *path-deviation distance* (PDD) between the actual robot coordinates and a straight line, which goes from the robot coordinates obtained when the last desired coordinates were given to these same coordinates. This distance corresponds to parameter "*R*" of the LMPC cost function in (8).
- The last parameter represents the changes allowed to the input signal and corresponds to parameter "*S*" of the LMPC cost function in (8).

One consideration that should be taken into account is the different distance magnitudes of P (APD), Q (OD) and R (PDD). The approaching distance could be more than one metre but the magnitude of the deviation distance is normally in the order of cm. This only becomes effective when the robot is approaching the final desired point. Hence, when a further reduction in the deviation distance to less than 1cm is attempted, an increase in the weight value for the deviation distance in the cost function is proposed. It is in this context that the APD factor has been normalized. In addition, the orientation deviation factor has to be analysed and normalized, especially if it is expressed in radians. The conditions imposed are that the orientation deviation factor has to be sensitive to the different errors in order to perform an effective motion control. PID and LMPC path-tracking strategies are similar. The desired path to be followed can be composed of a sequence of straight and turning actions. In this study, the cost function parameters are P=1, Q=1, R=1 and S=0. Control law tuning is carried out as a function of the kind of path being tracked (straight, wide left turn, slight left turn, wide right turn and slight right turn). A path is considered to be straight when the orientation error is less than 5°. A slight turn should be commanded when the left or right heading error is equal to or larger than 5° and less than 25°. Finally, a wide turn should be commanded when the orientation error is larger than or equal to 25°. It should be noted that the methodology is similar to the one that is implemented for PID controllers. Factor tuning of the APD (approaching-point distance), OD (orientation deviation) and PDD (path-deviation distance) parameters is done by considering

experimental data [22]. High and low values are set by slightly modifying the values that correspond to sensitive parameters and verifying their validity for a set of paths. Heuristic experiments for selecting high and low values of APD, OD and PDD have depicted a suitable interval up to 50%. Statistics for each path for parameters, such as time (T), path deviation (PD), travelled distance (TD) and averaged speed (AS), are obtained. Path-tracking performance is analysed using the means of the different factor weights (high or low value). For each combination of factors, three different runs were tried. The averaged value of the three runs makes a statistical analysis of each factor combination possible. From these standard deviations, the importance of the factor effects can be determined using a rough rule that considers the effects when the value differences are close to or greater than two or three times their standard deviations.

5. Comparing LMPC and PID Techniques for Path-Following

This section analyses the PID and LMPC control results that were obtained. Previous results are used to perform function factor adjustments with the aim of securing better control laws. LMPC and PID controllers are tested by using longer paths that include a set of the basic movements analysed in Section 3 and 4.

5.1 Analysis of PID and LMPC Results

The qualitative results in PID control and LMPC performances show the following:

- PID controllers usually spend less time performing the set of basic movements. However, some LMPC factor combinations are as time-effective as PID controllers.
- Except for wide left turn, path-error is less or similar when LMPC strategies are used. Moreover, LMPC performs with much more accuracy when straight path-following is analysed. It is noticeable that some LMPC factor combinations for wide left turn perform in a similar way to PID controllers.
- Distances are normally larger when PID controllers are used.
- In normal circumstances, the averaged speeds are higher for PID controllers. However, some factor combinations of LMPC can produce a similar performance to that of PID controllers.

Figure 6 shows the LMPC and PID results when path-following of the different five basic actions is performed.

LMPC usually performs with better path-error accuracy. Turning asymmetries can be caused by the differences between the right and left velocity models and are emphasized for wide turning. The analysis is performed by taking into consideration the different paths:

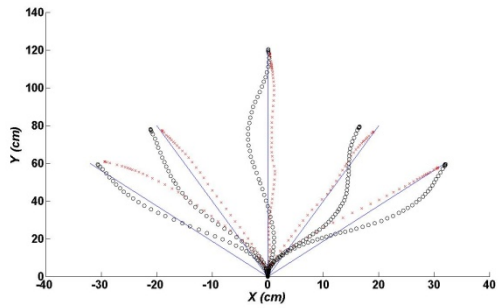

Figure 6. Path-following results for PID control (dots) and LMPC (crosses)

- There are not significant effects for PID control and LMPC strategies when straight line path-following is performed. However, with regard to LMPC, OD and PDD factors seem to reduce the averaged-speed with effects close to two times their standard deviations.
- An analysis of wide left turns for PID controllers reveals the lateral effects of time and averaged-speed. When high values for K_R and K_L factors are selected, the time is decreased, while the averaged-speed is increased. With LMPC, the effects of APD and OD reduce the necessary time, while PDD increases it. Path-error effects also happen with OD and PDD factors, with OD decreasing the error and PDD increasing it.
- The K_L factor effect reduces the distance travelled as far as slight left turns for PID controllers are concerned. When high values of K_R and K_L factors are selected, a reduction in the time and the distance travelled is accomplished simultaneously. LMPC effects for APD and OD factors reduce time, while increasing the averaged-speed. Otherwise, PDD factor increases the time and reduces the speed.
- An analysis of wide right turns for PID controllers shows an error reduction when K_R and K_L factors are set to high values. When the OD factor is considered, the LMPC effect reduces the time and increases the speed. Other PDD, OD and PDD effects increase the time and reduce the speed.
- A slight right turn analysis for PID controllers does not show any time effects. However, with regard to LMPC strategies, the PDD factor increases the time and reduces the path-error and averaged speed, and so do APD and OD factors.

Considering the qualitative results, it is suggested that a fixed control law for PID controllers is used when a shorter time and higher velocities are sought. Meanwhile, high values for K_R and K_L factors are proposed for the complete set of paths. Otherwise, the LMPC control law is proposed as a function of the path to be followed because this flexibility improves the results. Table 4 shows the different factors selected as a function of the path being tracked.

Type of Motion	Factor		
	APD	OD	PDD
Straight	H	L	L
Wide left turn	H	H	L
Wide right turn	H	H	L
Slight left turn	H	H	L
Slight right turn	L	H	L

Table 4. LMPC flexible factor values as function of the path

5.2 Testing PID and LMPC Using Longer Paths

In this subsection, the previously obtained control results are tested by commanding three different paths. These have to include the five different basic ones studied previously. In order to achieve this, a set of 12 waypoints with a similar shape to a dodecagon is proposed. The path is tested in clockwise and anticlockwise directions in order to test both right and left turns. A path that includes right and left turns and is composed of 17 waypoints is also analysed. The experiments are carried out for fixed PID and flexible LMPC controllers. From our lab experiences, by applying LMPC and PID algorithms, the WMR is able to perform changes of 33 degrees on the heading orientation within a distance of 60cm defined by two consecutive waypoints.

In order to analyse the accuracy, the sequence of points is commanded each time that the previous point is reached. It should be noted that, while this produces a reduction in the speed each time the WMR is close to the final commanded point, a better path-deviation accuracy is achieved. As explained in the previous sections, three different runs were tried for each control law and path, with the averaged value of the three runs. This makes a statistical analysis for each control law and path possible. The measured parameters are: time (T), path-deviation (PD), travelled distance (TD) and averaged speed (AS). Figure 7 shows the statistical results obtained for the clockwise dodecagon path when flexible LMPC and PID control laws are compared. Statistics of flexible LMPC and PID control laws, when differences between LMPC and PID means are analysed, depict a time effect of 2.9s (σ_T=1.6s), path-deviation error effect of -2.9cm (σ_{PD}=0.45cm), travelled distance effect of -57cm (σ_{TD}=4.4cm) and averaged speed effect of -3.3cm/s (σ_{AS}=0.85cm/s). By comparing LMPC and PID control performance, a small negative PID time effect with a value close to two times the standard deviation can be observed, and so can substantial positive PID effects that increase the path deviation, travelled distance and averaged speed. Figure 8 shows the path-following results obtained for PID and flexible LMPC.

An anticlockwise dodecagon is proposed for analysing left turn paths. Figure 9 shows the statistical results obtained when flexible LMPC and PID control laws are compared. Statistics of LMPC and PID control laws, when differences between LMPC and PID means are analysed, depict a time effect of -0.6s (σ_T=1.04s), path deviation effect of -4.2cm (σ_{PD}=0.13cm), travelled distance effect of -82cm (σ_{TD}=2.28cm) and averaged speed effect of -2.4cm/s (σ_{AS}=0.51cm/s). A comparison of LMPC and PID control laws shows that that negative PID time effects can be ignored. Important positive PID effects that increase path deviation, travelled distance and averaged speed can be seen. Figure 10 shows the path-following results obtained for PID and flexible LMPC.

Finally, right and left turns are analysed within a path that is similar to a figure-of-eight shape. Figure 11 depicts the statistical results obtained when flexible LMPC and PID control laws are compared.

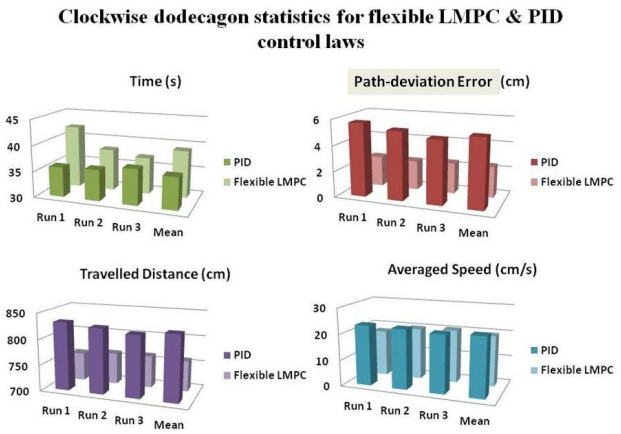

Figure 7. Clockwise dodecagon results for PID and LMPC

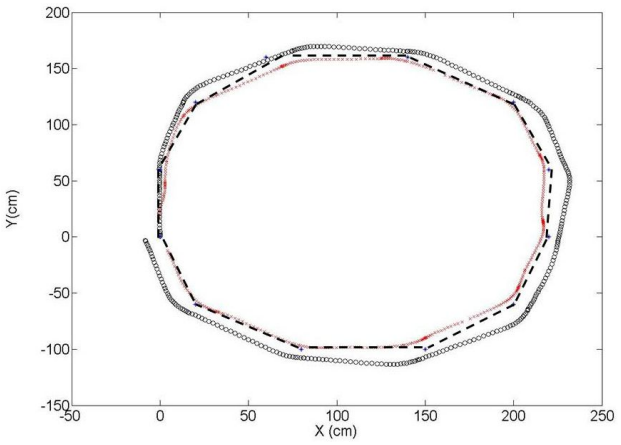

Figure 8. Clockwise dodecagon path-following results for PID (empty circles) and LMPC (crosses), compared to the desired path (dashes)

Statistics of LMPC and PID control laws, when differences between LMPC and PID means are analysed, depict a time effect of 1.4s (σ_T=1.2s), path deviation effect of -5.1cm (σ_{PD}=0.27cm), travelled distance effect of -141cm (σ_{TD}=3.7cm) and averaged speed effect of -2.9cm/s (σ_{AS}=0.38cm/s). An analysis of flexible LMPC and PID control laws shows substantial positive PID effects that increase path deviation, travelled distance and averaged speed. Figure 12 shows the path-following results obtained for PID and flexible LMPC. PID and LMPC techniques use the same set of reduced models, depicted in Table 2. From the results obtained when LMPC and PID are compared, we can see that the PID performance is poor. In this work, the use of a simple high level motion planning strategy with

parameters that are similar to those proposed for the LMPC cost function could produce unsatisfactory results. By using some complementary high level algorithms, which include some constraints and path-planning algorithms that are related to vehicle kinematics, it is expected to be able to improve the PID control performance.

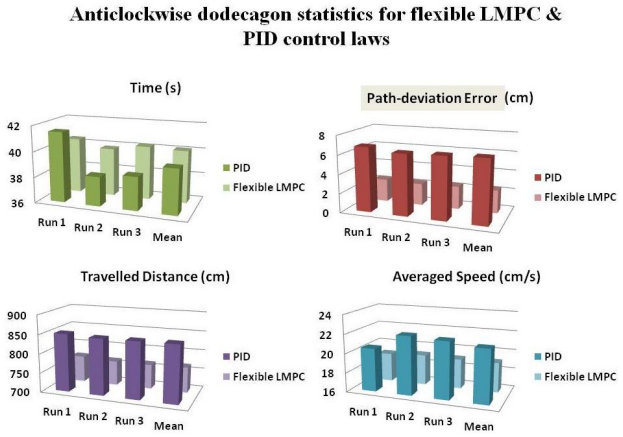

Figure 9. Anticlockwise dodecagon results for PID and LMPC

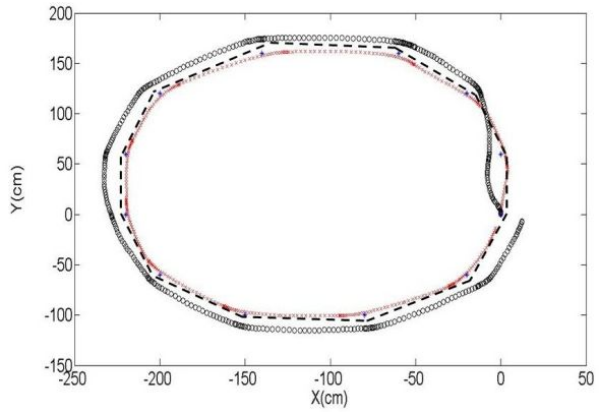

Figure 10. Anticlockwise dodecagon path-following results for PID (empty circles) and LMPC (crosses), compared to the desired path (dashes)

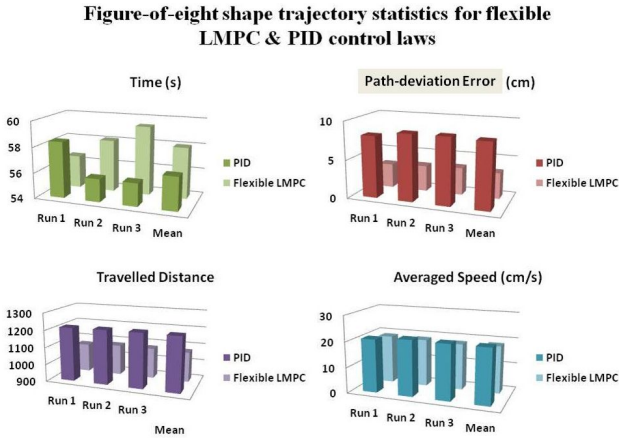

Figure 11. Eight-shape path analysis results for PID and LMPC

An analysis of the robustness was carried out experimentally by considering the reduced model parameters. Parametric uncertainty studies revealed a robust behaviour for model parameter variations of 20% of time-constant and 11% of gain.

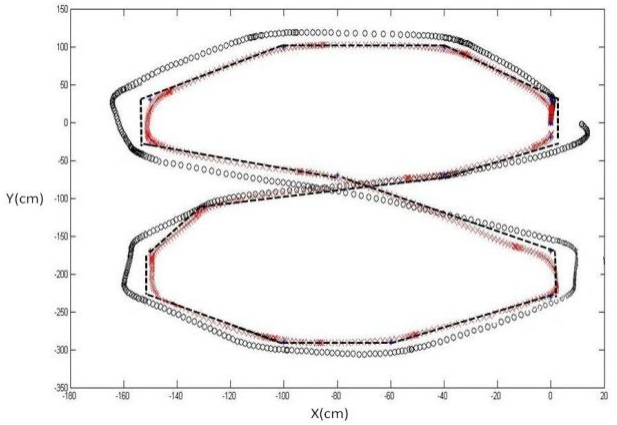

Figure 12. Figure-of-eight shape path results for PID (empty circles) and flexible LMPC (crosses), compared to the desired path (dashes)

6. Conclusions and Future Research

The results and methodology that are presented in this paper are also applicable to other robotic systems, in which the knowledge of the set of dynamic models that describe the system is required.

PID and LMPC strategies for path-following have been analysed. LMPC has been shown to be more time-expensive than traditional PID controllers. However, LMPC is not only based on PID speed-control approaches but also on coordinates that are available within short prediction horizons, which act as a reactive horizon. Other differences between the two methodologies are inherent in their control laws. PID controllers perform low level speed control, while LMPC strategies are based on selecting the closest path-following coordinates. Considering the implementation that is depicted in this research, LMPC performs path-following with less error, while PID controllers can make it faster. However, PID control can sometimes generate larger trajectory deviations, with a consequent increase in time. Motion planning, which includes kinematic constraints and/or some kind of path-planning, is a necessary layer for PID speed controllers. Instead of using two control layers, LMPC uses the horizon of achievable path coordinates as a cost function objective, making path-following easier to implement. In this way, it is expected that the difference of total processing time between PID and LMPC control strategies can become small when the path-planning processing time of PID controllers is considered. LMPC can also be a suitable methodology for other robotic systems, as, i.e., the case of multiple robots if the path-following algorithms can be performed in real time.

The LMPC cost function minimizes the path-deviation distance by searching for the best command speeds within the horizon of prediction. A single layer is enough for performing path-following and obtaining command speeds. LMPC is a suitable solution for low level path-following and has many interesting features when compared with traditional PID controllers. In this research, the horizon of prediction is set to half a second in order to deal with reactive behaviour. An analysis of fixed and flexible control laws for PID and LMPC methods reveals a significant improvement when using flexible LMPC control laws. Future research should be directed towards testing other motion planning strategies for PID controllers that reduce the total travelled distance of a WMR.

7. References

[1] Åström, K. J., and Hägglund, T. (2001) The Future of PID Control, *Control Engineering Practice*, 9:1167-1175.

[2] Maciejowski, J. M. (2002) *Predictive Control with Constraints*, Ed. Prentice Hall, Essex, UK.

[3] Qin S.J., and Badgwell, T.A., (2003) A Survey Of Industrial Model Predictive Control Technology, *Control Engineering Practice*, 11 (7): 733-764.

[4] Pacheco, L., and Luo, N. (2011) Mobile Robot Local Trajectory Tracking with Dynamic Model Predictive Control Techniques, *International Journal of Innovative Computing, Information and Control*, 6(7): 3457-3483.

[5] Bloch, A. M. (2003) Nonholonomic Mechanics and Control, *Interdisciplinary Applied Mathematics*, vol. 24, Springer, New York, USA.

[6] Sørdalen, O. J. and Canudas de Wit, C. (1993) Exponential Control Law for a Mobile Robot: Extension to Path Following, in: *IEEE Transaction on Robotics and Automation*, 9(6):837-842.

[7] Reeds J. A. and Shep, L. A. (1990) Optimal Paths for a Car That Goes Forwards and Backwards. *Pacific Journal of Mathematics,* vol. 145.

[8] Astolfi, A. (1996) Discontinuous Control of Nonholonomic Systems, *Systems and Control Letters,* 27:37-45.

[9] Michalek, M., Kozlowski, K., (2009) Motion Planning and Feedback Control for a Unicycle in a Way Point Following Task: The VFO Aproach. *International Journal of Aplied Mathematics and Computer Science*, 19 (4):533-545.

[10] Ogata, K. (2009) *Modern Control Engineering*, fifth edition, New Jersey, USA, Prentice Hall.

[11] Ögren, P. and Leonard, N. (2005) A Convergent Dynamic Window Aproach to Obstacle Avoidance, in: *IEEE Trans. Robotics*, 21(2):188-195.

[12] Matveev, A.S., Hoy, M., Katupitiya, J., Savkin, A.V. (2013) Nonlinear Sliding Mode Control of an Unmanned Agricultural Tractor in the Presence of Sliding and Control Saturation, *Robotics and Autonomous Systems*, 61(9):973-987.

[13] Genci Capi and Shin-ichiro Kaneko, (2009) Evolution of Neural Controllers in Real Mobile Robots for Task Switching Behaviors, *International Journal of Innovative Computing, Information and Control*, 5(11) (A):4017-4024.

[14] Cheng, Y., Wang X., and Lei, R. (2009) A Fuzzy Control System for Path Following of Mobile Robots, *ICIC Express Letters*, 3(3)(A):403-408.

[15] Pourboughrat, F., M., Karlsson, M. P. (2002) Adaptive Control of Dynamic Mobile Robots with Nonholomic Constraints, in: *Computers and Electrical Engineering*, 28:241-253.

[16] Primbs, J. A., Nevistic, V., and Doyle, J. C. (1999) Nonlinear Optimal Control: A Control Lyapunov Function and Receding Horizon Perspective, *in: Asian J. Control*, 2(1):14-24.

[17] Klancar, G., and Skrjanc, I. (2007) Tracking-error Model-based Predictive Control for Mobile Robots in Real Time, in: *Robotics and Autonomous Systems*, 55:460-469.

[18] Ollero, A., and Amidi, O., (1991) Predictive Path Tracking of Mobile Robots. Aplication to the CMU Navlab, in: *Proceedings of 5th International Conference on Advanced Robotics, Robots in Unstructured Environments*, 2:1081-1086.

[19] Kühne, F., Lages, W. F., Gomes da Silva Jr., J. M. (2005) Model Predictive Control of a Mobile Robot Using Input-output Linearization, in: *Proceeding of Mechatronics and Robotics*, 3:1163-1168.

[20] Darby, M. L., Nikolaou, M. (2012) MPC: Current Practice and Challenges, *Control Engineering Practice,* 20:328-342.

[21] Pacheco, L. and Luo, N. (2006) Mobile Robot Experimental Modeling and Control Strategies Using Sensor Fusion, in: *Control Engineering and Aplied Informatics*, 8(3):47-55.

[22] Box, G. E. P., Hunter, J. S., and Hunter, W. G., (2005) *Statistics for Experimenters*, Wiley Series in Probability and Statistics, D. J. Balding et al. (Eds.):New Jersey, USA.

[23] Ljung, L., (1987) *System Identification: Theory for the User,* Ed. Prentice Hall, Englewood. Cliffs, NJ.

[24] Norton, J. P. (1986) *An Introduction to Identification,* Ed. Academic Press, New York, 1986.

[25] Åström, K. J., and Wittenmark, B. (1997) Computer-Controlled Systems: Theory and Design (3rd Edition):Ed. Prentice Hall, NJ, USA.

[26] Noborio, H., and Schmidt, G. (1996) Mobile Robot Navigation Under Sensor and Localization Uncertainties. *Autonomous Mobile Systems*, Ed. Springer-Verlag, London, UK, p.118-127.

[27] Wan, J., (2007) *Computational Reliable Aproaches of Contractive MPC for Discrete*-time Systems, PhD Thesis, University of Girona.