Neural Network-Based Self-Tuning Kinematic Control and Dynamic Compensation for Mobile Robots

Luis F. Recalde[1][0000−0001−5296−7385], Bryan S. Guevara[1][0000−0002−7830−0498], Danny J. Zea[2][0000−0002−9919−0531], and Víctor H. Andaluz[1][0000−0002−8127−1595]

[1] Universidad De Las Fuerzas Armadas ESPE, Sangolqui, Ecuador
{lfrecalde1,bsguevara,vhandaluz1}@espe.edu.ec
[2] Escuela Superior Politécnica de Chimborazo ESPOCH, Riobamba, Ecuador
danny.zea@espoch.edu.ec

Abstract. This paper proposes a Neural Network-Based Self-Tuning Kinematic Controller and Dynamic Compensation for tracking trajectories, which applied e.g., in cases where Mobile Robots are subject to: continuous parametric changes; different trajectories and external disturbances where online gains tuning is a desirable choice. For this kind of controller, the kinematic and dynamic model was developed considering that, the Mobile Robot is conformed by differential platform and the operating point is not located at the center of wheel's axes. The Artificial Neural Networks estimates the states of the mobile robot while the gradient descent optimization algorithm adjust the controller gains that attain the smaller position tracking error, the dynamic model is used to compensate the velocity errors in the robot. Moreover, the stability of the proposed controller is demonstrated analytically. Finally, simulation results are given considering a Turtlebot3 mobile robot, real time experiments are implemented in the same mobile robot, where the tests are carried out to show the effectiveness of the controller in a real environment

Keywords: Neural network · Self tuning · Optimization · Mobile robots · Auto-tuning.

1 Introduction

The main function of modern robotics is the development of functional robots, which are used in structured or partially structured environments, e.g. in applications like welding, cutting and exploration missions [3, 1]. Modern robotics are not only apply in the previous applications, in recent years there has been a noticeable increase in the use of robots in industrial field which has been one of the main precursors of more sophisticated control systems[14, 20].

The mobile robots are widely used in these applications, so this kind of systems are subject to disturbances, slippage, white noise and sensor errors, so that it is very difficult to implement high precision controllers [17]. No matter

the changes in its dynamics or variations in operating conditions, the tasks must be performed with due precision, for example, in the case of load transportation, characteristics of the dynamics, center of mass and inertia are values that change when the robot is loaded. Then, to keep a good performance, the controller must be able to adapt while transporting different types of loads [17].

Some works present the design of controllers that are capable of adapting itself, e.g., in [4] present the use of a radial basis neural network (RBF-NN) for mobile robot dynamics approximation, [19] shows the use of neural networks are based on multilayer feed forward neural networks with back-propagation learning or more efficient variations of this algorithm researches [12, 10] present systems with a mix of neural networks and fuzzy control in which the training and rules of behavior are based on the desired state.

In this context, this paper proposed a Neural Network-Based Self-Tuning Kinematic Controller and Dynamic Compensation to solve the trajectory tracking problem for a mobile robot with continuous parametric changes and different desired trajectories. The Artificial Neural Networks are used to develop the Self Tuning Kinematic Control, through gradient descent and back-propagation algorithm, finally the dynamic compensation which guarantees compliance of the velocities in the real mobile robot, it is used to keep the point of interest on the desired trajectory. Simulation and real time experiments on an Turtlebot3 mobile robot are conducted, the communication was done through ROS [11] to show the effectiveness of the proposed controller.

Including the introduction and the summary, this article is organized as follows: Section 2 details the kinematic and dynamic model representation of the mobile robot. The controller design and the control scheme are presented in Section 3. The simulation and experimental results are presented in section 4. Finally the conclusions are presented in section 5.

2 General System Model of Mobile Robots

In this section the kinematic and dynamic model of the system is presented, the system is composed by a differential mobile robot, therefore a unique and general system is formed. The configuration of a mobile platform is defined by a vector $\mathbf{q}$ with n independent coordinates, called generalized coordinates, where $\mathbf{q} = \begin{bmatrix} q_1 \; q_2 \; ... \; q_n \end{bmatrix}^\intercal = \begin{bmatrix} \mathbf{q_p} \end{bmatrix}^\intercal$ and $\mathbf{q_p}$ represents the generalized coordinates of the mobile platform, it is appreciated that $n = n_p$, where n_p are the configuration coordinates of the mobile platform, the set of these is denoted by $\mathcal{N}$. The position and orientation of the point of interest is defined by a vector $\boldsymbol{\eta}$ given by $\boldsymbol{\eta} = \begin{bmatrix} \eta_1 \; \eta_2 \; ... \; \eta_m \end{bmatrix}^\intercal = \begin{bmatrix} \boldsymbol{\eta_p} \end{bmatrix}^\intercal$ in $\mathbf{R}$ of $\boldsymbol{\eta_p}$ operational coordinates of the mobile platform, in the same way the set of these is denoted by $\mathcal{M}$[8].

2.1 Mobile Robot Kinematic Model

The Figure 1 shows a differential robot, where $\boldsymbol{\eta} = \begin{bmatrix} x \; y \; \psi \end{bmatrix}^\intercal$ is the position and orientation of the point of interest with respect to the global reference system

$< \mathcal{R} >$, where a is the distance between the geometric center and the wheel axis. The kinematic model in the mobile robot provides the derivative of the point of interest, the velocity can be represented as follows,

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} \cos(\psi) & -a\sin(\psi) \\ \sin(\psi) & a\cos(\psi) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mu \\ \dot{\psi} \end{bmatrix} \tag{1}$$

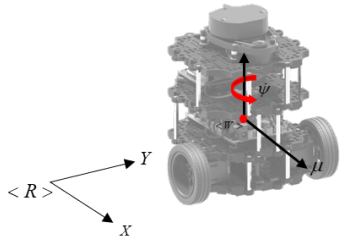

Fig. 1. Frame coordinates of Differential Mobile Robots

The equations (1) can be represented in a compact form (2), where $\dot{\boldsymbol{\eta}} = \begin{bmatrix} \dot{x} & \dot{y} & \dot{\psi} \end{bmatrix}^\intercal$ represents the velocities of the point of interest in the coordinate system $< \mathcal{R} >$, $\mathbf{J}(\psi)$ is the transformation matrix and $\mathbf{v}(t) = \begin{bmatrix} \mu & \dot{\psi} \end{bmatrix}^\intercal$ is the velocity vector of linear and angular velocities of differential robot [2].

$$\dot{\boldsymbol{\eta}}(t) = \mathbf{J}(\psi(t))\mathbf{v}(t) \tag{2}$$

2.2 Mobile Robots Dynamic Model

To obtain the dynamic model of the system, the Euler Lagrange equations are used, which propose a difference between kinetics and potential energy according to (3), and finally apply (4) [18, 15, 5].

$$\mathcal{L} = \mathcal{K} - \mathcal{P} \tag{3}$$

$$\frac{d}{dt}\left(\frac{d\mathcal{L}}{\dot{\boldsymbol{\eta}}}\right) - \frac{d\mathcal{L}}{\boldsymbol{\eta}} = f_i(t) \tag{4}$$

Kinetic energy is calculated from $\mathcal{K} = \frac{1}{2}\dot{\boldsymbol{\eta}}^\intercal m\ \dot{\boldsymbol{\eta}}$ and the potential energy is given by $\mathcal{P} = 0$, m represent the mass of the mobile platform and f_i are the forces in each coordinate system, in this sense the dynamic model shows in (5).

$$\begin{bmatrix} fx \\ fy \\ \tau\psi \end{bmatrix} = \begin{bmatrix} m & 0 & -am\sin(\psi) \\ 0 & m & am\cos(\psi) \\ -am\sin(\psi) & am\cos(\psi) & m(a^2+1) \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{\psi} \end{bmatrix} \begin{bmatrix} 0 & 0 & -am\cos(\psi)\dot{\psi} \\ 0 & 0 & -am\sin(\psi)\dot{\psi} \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \end{bmatrix} \tag{5}$$

The above equation can be expressed in matrix form, as

$$\mathbf{f}(t) = \mathbf{M}(\boldsymbol{\eta})\ddot{\boldsymbol{\eta}}(t) + \mathbf{C}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}})\dot{\boldsymbol{\eta}} \tag{6}$$

It is important to mention that $\mathbf{M}(\boldsymbol{\eta})$ is the inertia matrix, $\mathbf{C}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}})$ is the centripetal matrix and $\ddot{\boldsymbol{\eta}}(t)$, $\dot{\boldsymbol{\eta}}(t)$ are the acceleration and velocity vectors, respectively, with respect to the inertial frame $< \mathcal{R} >$. Turtlebot3 and others commercial robots have low level PID controllers, which does not allow controlling the motors directly using torque signals, therefore it is considered $\mathbf{B}(\psi)$, $\mathbf{D}$ and $\mathbf{E}$ are matrices that allow the transformation of the dynamic model. Simply PD servo controllers are considered to control each motor, $\mathbf{L}$ and $\mathbf{S}$ are constants gain matrices (7).

$$\mathbf{f}(t) = \mathbf{B}(\psi)[\mathbf{D}[\mathbf{L}\mathbf{v}_{\mathbf{ref}}(t) - \mathbf{L}\mathbf{v}(t) - \mathbf{S}\dot{\mathbf{v}}(t)] - \mathbf{E}\mathbf{v}(t)] \tag{7}$$

Finally, from equations (6) and (7) we obtain the dynamic model of the mobile robot, regarding as control signals of the mobile robot (8).

$$\begin{bmatrix} \mu_{ref} \\ \dot{\psi}_{ref} \end{bmatrix} = \begin{bmatrix} \chi_1 & 0 \\ 0 & \chi_2 \end{bmatrix} \begin{bmatrix} \dot{\mu} \\ \ddot{\psi} \end{bmatrix} + \begin{bmatrix} \chi_3 & \chi_4 + \chi_5\dot{\psi} \\ \chi_6\dot{\psi} & \chi_7 \end{bmatrix} \begin{bmatrix} \mu \\ \dot{\psi} \end{bmatrix} \tag{8}$$

The equations (8) can be represented in compact form (9), the dynamic model is obtained, where $\bar{\mathbf{M}}(\boldsymbol{\chi})$ represents the differential robot system's inertia, $\bar{\mathbf{C}}(\boldsymbol{\chi}, \mathbf{v})$ represents the components of the centripetal forces, $\mathbf{v}_{\mathbf{ref}}(t) = \begin{bmatrix} \mu_{ref} & \dot{\psi}_{ref} \end{bmatrix}^\intercal$ is a vector of reference velocities, $\mathbf{v}(t) = \begin{bmatrix} \mu & \dot{\psi} \end{bmatrix}^\intercal$ the real velocities of the robot, finally $\boldsymbol{\chi}$ is the vector of dynamic parameters, which contain the physical parameters of differential mobile robot.

$$\mathbf{v}_{\mathbf{ref}}(t) = \bar{\mathbf{M}}(\boldsymbol{\chi})\dot{\mathbf{v}}(t) + \bar{\mathbf{C}}(\boldsymbol{\chi}, \mathbf{v}(t))\mathbf{v}(t) \tag{9}$$

The identification and validation of the proposed dynamic model is show in Figure 2. The identification of the mobile robot system was carried out using optimization techniques, where an objective is to minimize a cost function conformed by a vector of error between real values of the Turtlebot3 velocities and the mathematical model [16], varying the values of the vector $\boldsymbol{\chi}$.

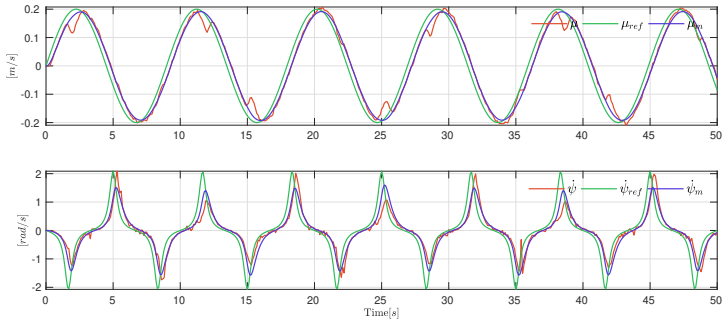

Fig. 2. Validation of the proposed dynamic model

3 Controller Design

The proposed control allows the point of interest of differential mobile robot track different trajectories. The control objective is achieved through a designed cascade control system, which comprises three stages: a kinematic controller generating the control actions for the tracking paths, the artificial neural networks are used to develop the kinematic self-tuning control, through estimation of the states of the mobile robot, the gradient descent optimization algorithm adjusts the controller gains online achieving the smallest position tracking error and a dynamic compensation that guarantees compliance with said velocities in the real robot, finally Figure 3 presents the proposed control scheme.

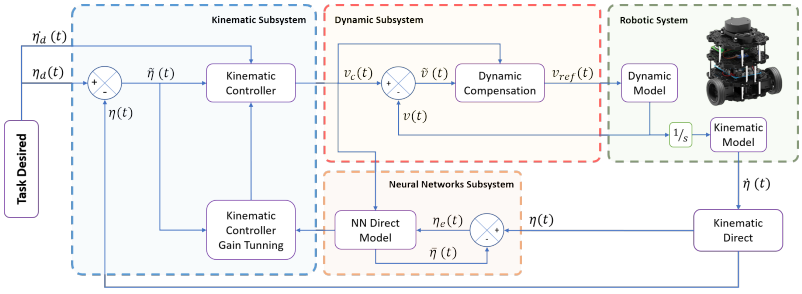

Fig. 3. Proposed Control Scheme.

3.1 Kinematic Controller

It is proposed inverse kinematic control (10) which allows finding the velocities for the differential mobile robots that allow tracking of the desired trajectory $\boldsymbol{\eta_d}(t) = \begin{bmatrix} x_d & y_d \end{bmatrix}$, where $\dot{\boldsymbol{\eta}}_{\mathbf{d}}(t) = \begin{bmatrix} \dot{x}_d & \dot{y}_d \end{bmatrix}$ are the velocities of the desired trajectory, $\tilde{\boldsymbol{\eta}}(t) = \begin{bmatrix} \tilde{\eta}_x & \tilde{\eta}_y \end{bmatrix}$ are the control errors that are obtained from

$\tilde{\boldsymbol{\eta}}(t) = \boldsymbol{\eta_d}(t) - \boldsymbol{\eta}(t)$, $\mathbf{J}^{-1}(\psi)$ is the inverse of the transformation matrix, $\mathbf{K_1}$ is a diagonal matrix that allows quantify control errors, $\mathbf{K_2}$ is a diagonal matrix which is used to delimit the reference velocities, finally $\mathbf{v_c}(t) = \begin{bmatrix}\mu_c & \dot{\psi}_c\end{bmatrix}^\intercal$ are the reference velocities generated by the controller.

$$\mathbf{v_c}(t) = \mathbf{J}^{-1}(\psi)(\dot{\boldsymbol{\eta}}_\mathbf{d}(t) + \mathbf{K_2}\tanh\left(\mathbf{K_2^{-1}}\mathbf{K_1}\tilde{\boldsymbol{\eta}}(t)\right)) \tag{10}$$

Equals equations (10) and (2) we get (11).

$$\mathbf{J}^{-1}(\psi)\dot{\boldsymbol{\eta}}(t) = \mathbf{J}^{-1}(\psi)(\dot{\boldsymbol{\eta}}_\mathbf{d}(t) + \mathbf{K_2}\tanh\left(\mathbf{K_2^{-1}}\mathbf{K_1}\tilde{\boldsymbol{\eta}}(t)\right)) \tag{11}$$

This equation is known as a closed-loop representation and considering the velocities error as $\dot{\tilde{\boldsymbol{\eta}}}(t) = \dot{\boldsymbol{\eta}}_\mathbf{d}(t) - \dot{\boldsymbol{\eta}}(t)$,therefore, the final expression would remain expressed as follows (12), this equation is useful to demonstrate the stability of the controller.

$$\dot{\tilde{\boldsymbol{\eta}}}(t) = -\mathbf{K_2}\tanh\left(\mathbf{K_2^{-1}}\mathbf{K_1}\tilde{\boldsymbol{\eta}}(t)\right) \tag{12}$$

3.2 Self-Tuning Kinematic Controller Gains

In controller based on the inverse kinematics given in equation (10), it can be noted that this algorithm presents a matrix of diagonal values $\mathbf{K_1}$,which presents the following structure (13), where K_x is a value proportional to the errors on the x axis, K_y is a value proportional to the errors on the y axis, the disadvantage of this algorithm is that the values must be chosen with care, since they vary depending on the desired trajectories and parametric changes.

$$\mathbf{K_1} = \begin{bmatrix} K_x & 0 \\ 0 & K_y \end{bmatrix} \tag{13}$$

Therefore, a kinematic self-tuning algorithm based on optimization according to the gradient descent method [6]. The mathematical structure that will be used to adapt the gains will be subject to optimization problems and the equation is given as,

$$\mathbf{G} = \frac{1}{2}\tilde{\boldsymbol{\eta}}^\intercal\ \mathbf{R}\ \tilde{\boldsymbol{\eta}} \tag{14}$$

where $\mathbf{G}$ is the cost function and $\mathbf{R}$ is positive definite diagonal matrix that will weigh the position errors. To find the gains that minimize the cost function, the chain rule method will be used, considering that, for this method to be applied, a vector $\boldsymbol{\kappa} = \begin{bmatrix}K_x & K_y\end{bmatrix}$ is generated, therefore the partial derivative with respect to $\boldsymbol{\kappa}$ is obtained (15).

$$\frac{\partial \mathbf{G}}{\partial \boldsymbol{\kappa}} = \tilde{\boldsymbol{\eta}}^\intercal\ \mathbf{R}\ \frac{\partial \tilde{\boldsymbol{\eta}}}{\partial \boldsymbol{\kappa}} \tag{15}$$

$$\frac{\partial \tilde{\boldsymbol{\eta}}}{\partial \boldsymbol{\kappa}} = -\frac{\partial \boldsymbol{\eta}}{\partial \mathbf{v_c}}\ \frac{\partial \mathbf{v_c}}{\partial \boldsymbol{\kappa}} \tag{16}$$

The first derivative in the above equation (16) can be defined as the Jacobian matrix with the system velocity inputs, which can be calculated using direct neural network model and back-propagation algorithm [9].

The above deduced that the controller's profits change and adapt by making the cost function tend to zero, therefore according to the algorithm of the descent of the gradient which is shown below represented in (17)

$$\Delta\boldsymbol{\kappa} = -\boldsymbol{\alpha}\,\frac{\partial \mathbf{G}}{\partial \boldsymbol{\kappa}} = \boldsymbol{\alpha}\;\tilde{\boldsymbol{\eta}}^{\intercal}\;\mathbf{R}\;\frac{\partial \boldsymbol{\eta}}{\partial \mathbf{v_c}}\;\frac{\partial \mathbf{v_c}}{\partial \boldsymbol{\kappa}} \tag{17}$$

where $\boldsymbol{\alpha}$ is a diagonal matrix with the learning rates of the gradient descent algorithm.

3.3 Artificial Neural Network Direct Model

The artificial neural network which is used to approximate the robot platform is show in Figure 4. The algorithm used to obtain is back-propagation method, chosen for its ability to adapt to changing environments [7].

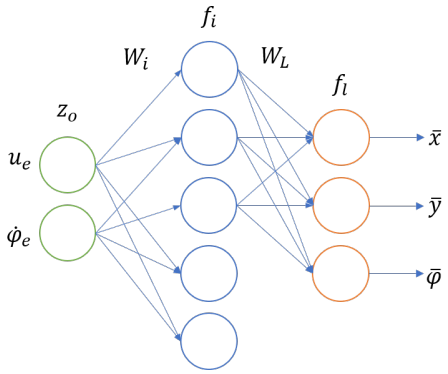

Fig. 4. Structure of the Neural Network

For the implementation of the back-propagation algorithm it is considered that neural networks has $\mathbf{L}$ layers in the internal structure, where $\mathbf{Z_0}$ is the input vector conformed by $\mathbf{Z_0} = \left[\mu_c\ \dot{\psi}_c\right]^{\intercal}$, $\mathbf{Z_L} = \left[\bar{x}\ \bar{y}\right]^{\intercal}$ is the output vector and $\boldsymbol{\eta} = \left[x\ y\right]^{\intercal}$ is the vector of the truth that the neural network tries to estimate. The weight matrices of the neural network are $\mathbf{W_1}, \mathbf{W_2}, ...\mathbf{W_L}$ and the non-linear activation's function's are $f_1, f_2, ...f_L$.

With these considerations forward propagation in the general form results in (18), the difference between the desired estimation and the values of the neural network is defined in (19) known as the error vector .

$$\mathbf{Z_i} = f_i(\mathbf{W_i}\mathbf{Z_{i-1}}) \tag{18}$$

$$\mathbf{E} = \left\|\mathbf{Z_L} - \boldsymbol{\eta}\right\|^2 \tag{19}$$

The back-propagation of the error vector in the general form are (20) and (21). Finally adaptive weights of the neural network are (22) and (23).

$$\delta_L = (\mathbf{Z_L} - \boldsymbol{\eta}) \otimes f_L^{'}(\mathbf{W_L Z_{L-1}}) \tag{20}$$

$$\delta_i = \mathbf{W}_{i+1}^{\intercal}\delta_{i+1} \otimes f_i^{'}(\mathbf{W_i Z_{i-1}}) \tag{21}$$

$$\frac{\partial \boldsymbol{E}}{\partial \mathbf{W_i}} = \delta_i \mathbf{Z}_{\mathbf{i-1}}^{\intercal} \tag{22}$$

$$\mathbf{W_i} = \mathbf{W_i} - \alpha \otimes \frac{\partial \boldsymbol{E}}{\partial \mathbf{W_i}} \tag{23}$$

The Jacobian matrix that can be found from the above neural network is (24), where $\mathbf{I}$ is a identity matrix.

$$\frac{\partial \bar{\boldsymbol{\eta}}}{\partial \mathbf{v_c}} = \mathbf{W}_{i+1}\ \mathbf{I} \cdot (\mathbf{I}^{\intercal} \otimes f_i^{'}(\mathbf{W_i Z_{i-1}}))\mathbf{W}_i \tag{24}$$

3.4 Dynamic Compensation

The differential Mobil robot presents dynamic effects, so velocities errors are generated $\tilde{\mathbf{v}} = \mathbf{v_c} - \mathbf{v}$, which is the difference between the velocities of the Kinematics controller and the real velocities of the robot. To reduce this effect, a dynamic compensation is proposed which is performed using the dynamic model of the robot.

The control law to be used is indicated in (25), where $\mathbf{K_3}$ it is a constant diagonal matrix proportional to the velocities errors, $\mathbf{K_4}$ it is a diagonal matrix that delimits the velocities sent to the mobile robot.

$$\mathbf{v_{ref}} = \bar{\mathbf{M}}(\mathbf{q})(\dot{\mathbf{v}}_\mathbf{c} + \mathbf{K_4} \tanh{(\mathbf{K_4^{-1}} \mathbf{K_3} \tilde{\mathbf{v}})}) + \bar{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}})\mathbf{v_c} \tag{25}$$

Grouping (25), (9) and considering the acceleration errors $\dot{\tilde{\mathbf{v}}}(t) = \dot{\mathbf{v_c}}(t) - \dot{\mathbf{v}}(t)$ the equation (26) is deduced.

$$\dot{\tilde{\mathbf{v}}} = -\mathbf{K_4} \tanh{(\mathbf{K_4^{-1}} \mathbf{K_3} \tilde{\mathbf{v}}(\mathbf{t}))} \tag{26}$$

3.5 Stability Analysis of Kinematic Controller

For the stability analysis, the Lyapunov method is used, starting with the candidate function of quadratic position errors (27) and applying the temporal derivative is obtained (28).

$$\mathbf{V}(\tilde{\boldsymbol{\eta}}) = \frac{1}{2}\tilde{\boldsymbol{\eta}}(t)^{\intercal}\tilde{\boldsymbol{\eta}}(t) \tag{27}$$

$$\dot{\mathbf{V}}(\tilde{\boldsymbol{\eta}}) = \tilde{\boldsymbol{\eta}}(t)^{\intercal}\dot{\tilde{\boldsymbol{\eta}}}(t) \tag{28}$$

By means of (12) and (28) is determined.

$$\dot{\mathbf{V}}(\tilde{\boldsymbol{\eta}}) = -\tilde{\boldsymbol{\eta}}(t)^{\intercal}\mathbf{K_2} \tanh{(\mathbf{K_2^{-1}} \mathbf{K_1} \tilde{\boldsymbol{\eta}}(t))} \tag{29}$$

Guaranteeing the stability of the kinematic controller if $\mathbf{K_1} > 0$ and $\mathbf{K_2} > 0$, so that $\tilde{\boldsymbol{\eta}} \to 0$ when $t \to \infty$.

3.6 Stability Analysis of Dynamic Compensation

For the stability analysis, the Lyapunov method is used (30), starting with the candidate function of quadratic velocities errors and applying the temporal derivative is obtained (31).

$$\mathbf{V}(\tilde{\mathbf{v}}) = \frac{1}{2}\tilde{\mathbf{v}}(t)^{\intercal}\tilde{\mathbf{v}}(t) \tag{30}$$

$$\dot{\mathbf{V}}(\tilde{\mathbf{v}}) = \tilde{\mathbf{v}}(t)^{\intercal}\dot{\tilde{\mathbf{v}}}(t) \tag{31}$$

By means (31) and (26), (32) is obtained.

$$\dot{\mathbf{V}}(\tilde{\mathbf{v}}) = -\tilde{\mathbf{v}}^{\intercal}(t)\mathbf{K_4}\tanh\left(\mathbf{K_4^{-1}}\mathbf{K_3}\tilde{\mathbf{v}}(\mathbf{t})\right) \tag{32}$$

Guaranteeing the stability of the dynamic compensation if $\mathbf{K_3} > 0$ and $\mathbf{K_4} > 0$, so that $\tilde{\mathbf{v}} \to 0$ when $t \to \infty$.

4 Results

4.1 Simulation Results

The validation of the proposal controller is carried out through the Webots simulation software [13]. The modeling of the structures and the simulation scenario is generated in the Sketch CAD software, then imported to the Webots graphics engine. The simulation scenario has a circular area with a radius of 3 meters, as shown in Figure 5.

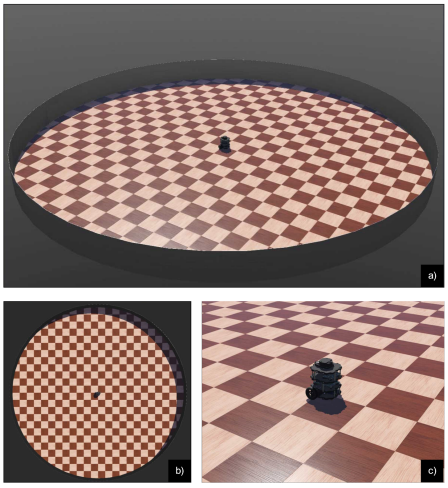

Fig. 5. 3D simulation scenario. (**a**) isometric view. (**b**) aerial view (**c**) Turtlebot3 mobile robot.

The collision system, the force of gravity and the coefficient of friction between contact surfaces are included in the physics engine of the simulator. The shadows and the simulation are configured in the rendering at every instant of time. The virtual stage configuration in the scene tree is shown in the Table 1.

Table 1. Webots World Configuration.

Field Scenario	Values
Gravity (x, y, z)	$(0, -9.81, 0)$
Constraint Force Mixing	0.00005
Error reduction parameter	0.2
Basic Time Step	$64[ms]$
Frames per second	60
Physics Disable time	$1[s]$
Physics Disable Linear Threshold	$0.01[m/s]$
Physics Disable Angular Threshold	$0.01[rad/s]$
North Direction	$(1, 0, 0)$
Gps reference	$(0, 0, 0)$

The Webots simulation software includes CAD models of different types of robots, among which is the Turtlebot3 mobile robot Figure 6. The simulated robot is characterized with the same configuration as a real robot.

Fig. 6. Turtlebot3 mobile robot Webots

The reading of the sensors and the actions of the different states of speed, acceleration, among others, are possible due to the libraries that can be programmed in a controller within the Webots simulation software. The Turtlebot3

mobile robot integrates different sensors, such as a reference GPS and a gyroscope that allow knowing the location and orientation respectively. The global and local reference system are configured with the direction of their axes according to the law of the right hand.

Following the scheme shown in Figure 7, the connection of the robot is established through ROS nodes, which allow direct communication between the data of the robot states and the mathematical software that implements the control algorithms. A full Simulation and Hardware in the Loop scheme is established for the evaluation of the implemented controller.

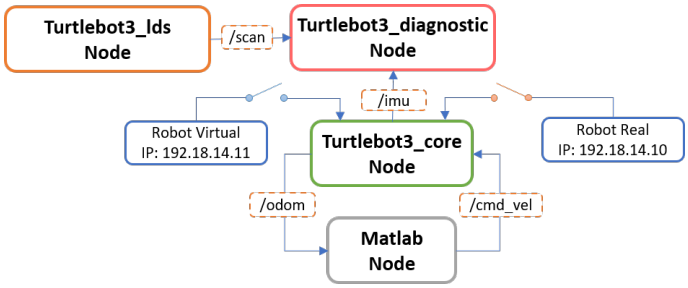

Fig. 7. Communication structure

Consists of following a desired trajectory while analyzing the behavior of Self Tuning Kinematic controllers under the disturbances, slippage and white noise generated by Webots. The desired trajectory and the initial conditions for the experiment are presented in the Table 2.

Table 2. Desired trajectory and initial configuration for Simulation

Parameters	Values	Parameters	Values
x_d	$0.5\cos(0.3t)[m]$	x_o	$0.1[m]$
y_d	$0.5\sin(0.3t)[m]$	y_o	$0.0[m]$

The gains for the Simulation are defined in Table 3, considering that an initial values of $\mathbf{K_1}$ is established and $\mathbf{K_2}$ was established under considerations of the limitations of real control velocities.

Figure 8 indicates the behavior of the Turtlebot3 robot in simulator Webots, it can be visualized that the point of interest tends to the established trajectory in Table 2. Figure 9 It shows the evolution of the control errors and it is appreciated that it tends to zero despite the white noise that is introduced into the Simulator.

Table 3. Initial values for controllers in Simulation

Parameters	Values	Parameters	Values
$\mathbf{K_1}$	$diag[0.5\ 0.5]$	$\mathbf{K_2}$	$diag[0.2\ 0.2]$
$\mathbf{K_3}$	$diag[1\ 1]$	$\mathbf{K_4}$	$diag[1\ 1]$
$\mathbf{R}$	$diag[1\ 1]$	$\boldsymbol{\alpha}$	$diag[0.1\ 0.1]$

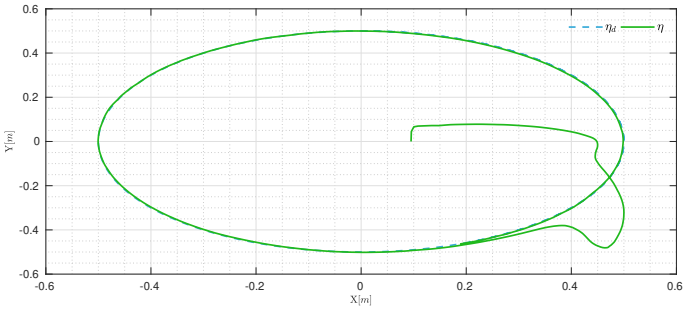

Fig. 8. Movement of the Turtlebot3 in Simulation.

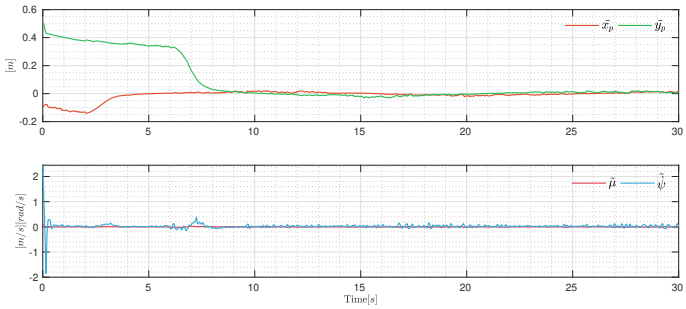

Fig. 9. Control Errors in Simulation

Figure 10 shows the kinematic and dynamic control values applied to the mobile robot during the simulation.

Figure 11 indicates the estimations of the states in the robot during the simulation, this values are really important for the adaptation of the gains in the kinematic controller. Figure 12 shows the evolution of the kinematic controller gains.

4.2 Experimental Results

The experimental test was implement on a TurtleBot3 robot shows in Figure 13, which admits linear an angular velocities as input reference signals, the controllers were carried out in Matlab software and the communication was done

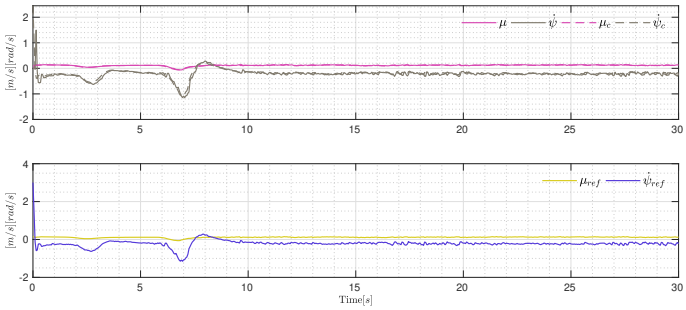

Fig. 10. Control Values of the Simulation.

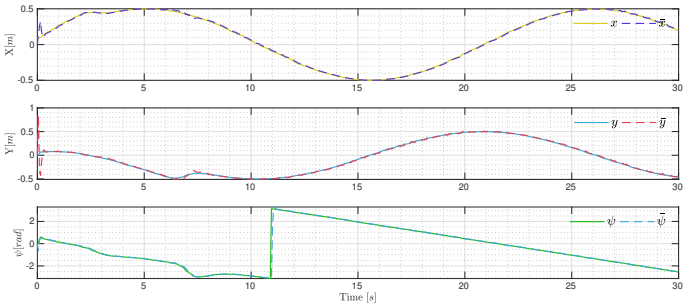

Fig. 11. Estimations of NN Direct Model of the Simulation

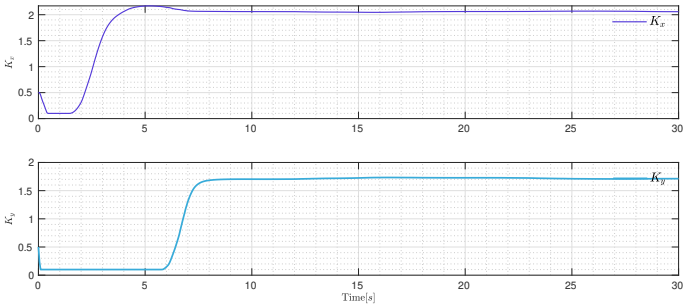

Fig. 12. Adaptive Gains Evolution of the Simulation

through robotic operating system ROS and the same communication structure shown in Figure 7.

The desired trajectory and the initial conditions for the experiment are presented in the Table 4. The gains for the Experiment are the same as the Simulation, defined in Table 3.

Figure 14 indicates the behavior of the Turtlebot3 robot in a real environment, it is clearly to see that the evolution of the point of interest is tracking the desired trajectory.

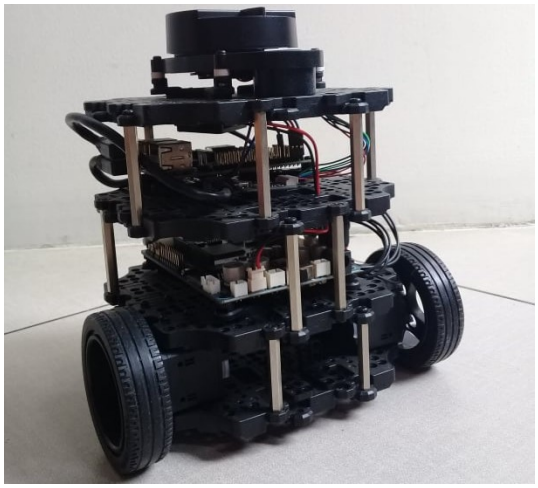

Fig. 13. Real Turtlebot3 mobile robot

Table 4. Desired trajectory and initial configuration for Experiment

Parameters	Values	Parameters	Values
x_d	$0.4\cos(0.3t)[m]$	x_o	$0.1[m]$
y_d	$0.4\sin(0.3t)[m]$	y_o	$0.0[m]$

Figure 15 shows the evolution of the control errors and it is appreciate tends to zero.

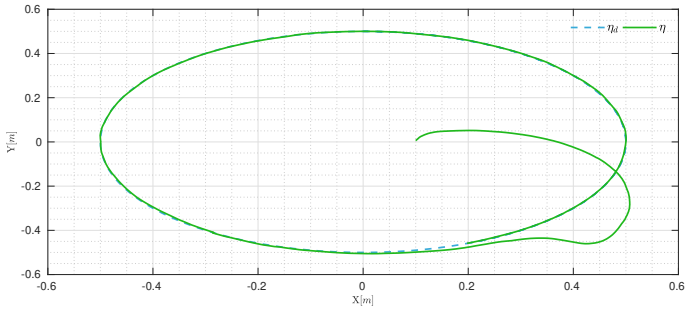

Fig. 14. Movement of the Turtlebot3 in Experiment.

The control errors of the dynamic compensation also tend to zero because the controller maintains the values generated by a kinematic controller in the mobile robot.

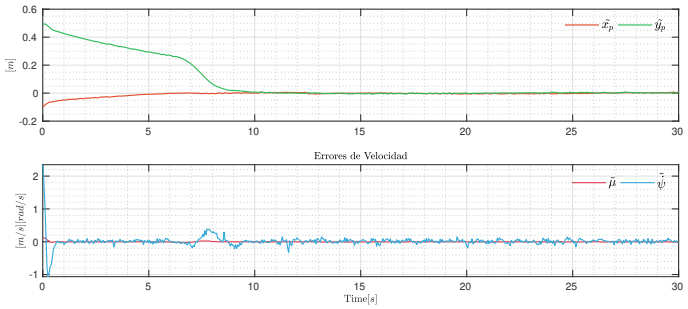

Fig. 15. Control Errors in Experiment

Figure 16 shows the kinematic control values applied to the mobile robot during the experiment. It is clear to see that the dynamic compensation generates the appropriate control values for compensate the dynamics in the real robot, guaranteeing the convergence to zero of the control errors.

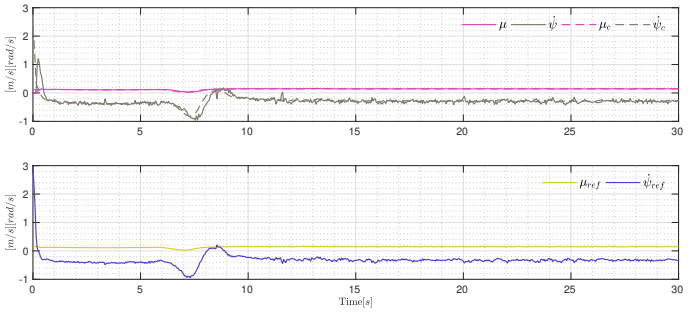

Fig. 16. Control Values of the Experiment.

Figure 17 indicates that the NN Direct Model makes a correct estimate of the positions and orientations of the mobile robot, which is important when adjusting the controller gains.

Figure 18 shows the evolution of the kinematic controller gains, which adapt considering the desired trajectory and the estimations of the states of the robot in a real environment.

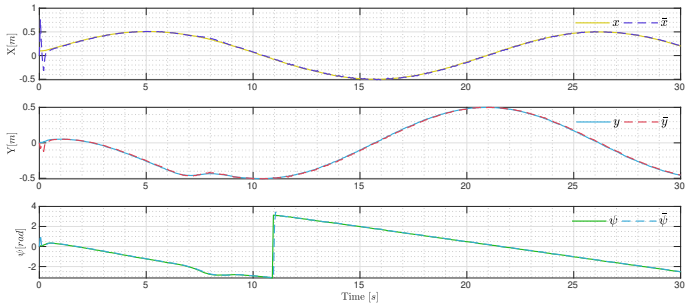

Fig. 17. Estimations of NN Direct Model of the Experiment

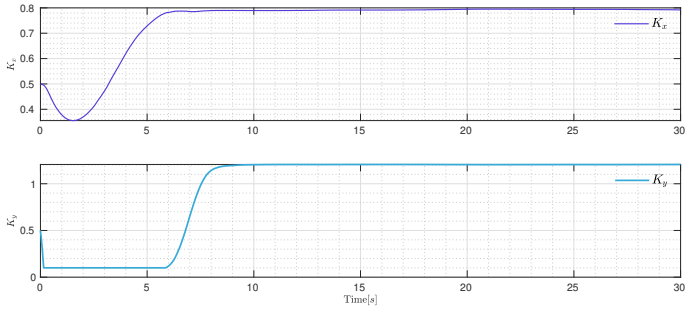

Fig. 18. Adaptive Gains Evolution of the Experiment

5 Conclusions

In this work the design of a Neural Network-Based Self-Tuning Kinematic Controller and Dynamic Compensation for tracking trajectories, was presented. The design of this controller is based on the kinematic and dynamic model of a differential robot, the Artificial Neural Networks estimates the states of the mobile robot while the gradient descent optimization algorithm adjusts online the controller gains. The performance was evaluated in two instances: simulation in Webots with a Turtlebot3 robot and a real Turtlebot3 in real time. In Webots simulation took place with white noise in the measurements of the GPS, including disturbances of changes of mass of the robot. The second validation was implement in a real Turtlebot3 in order to validate the controller in a real time experiment, both experiments shows a good performance of the proposed controller.

Acknowledgements The authors would like to thank Proyecto de Investigacion: Análisis, diseño e implementación de algoritmos de control inteligente en controladores con una red de sensores IoT en vehículos para mejorar la seguridad vial; Escuela Superior Politécnica de Chimborazo ESPOCH and the Research Group GITEA, for the support for the development of this work.

References

1. Andaluz, G.M., Andaluz, V.H., Terán, H.C., Arteaga, O., Chicaiza, F.A., Varela, J., Ortiz, J.S., Pérez, F., Rivas, D., Sánchez, J.S., Canseco, P.: Modeling dynamic of the human-wheelchair system applied to NMPC. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). vol. 9835 LNCS, pp. 179–190. Springer Verlag (2016). https://doi.org/10.1007/978-3-319-43518-3_18
2. Andaluz, V., Roberti, F., Carelli, R.: Robust control with redundancy resolution and dynamic compensation for mobile manipulators. In: Proceedings of the IEEE International Conference on Industrial Technology. pp. 1469–1474 (2010). https://doi.org/10.1109/ICIT.2010.5472488
3. Andaluz, V.H., Canseco, P., Varela Aldas, J., Ortiz, J.S., Pérez, M.G., Roberti, F., Carelli, R.: Robust control with dynamic compensation for human-wheelchair system. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) **8917**, 376–389 (2014). https://doi.org/10.1007/978-3-319-13966-1_37
4. Bugeja, M.K., Fabri, S.G.: Dual adaptive control for trajectory tracking of mobile robots. In: Proceedings - IEEE International Conference on Robotics and Automation. pp. 2215–2220 (2007). https://doi.org/10.1109/ROBOT.2007.363649
5. De La Cruz, C., Carelli, R.: Dynamic model based formation control and obstacle avoidance of multi-robot systems. Robotica **26**(3), 345–356 (5 2008). https://doi.org/10.1017/S0263574707004092, `https://www.cambridge.org/core/product/identifier/S0263574707004092/type/journal_article`
6. Fierro, R., Lewis, F.L.: Control of a nonholonomic mobile robot using neural networks. IEEE Transactions on Neural Networks **9**(4), 589–600 (1998). https://doi.org/10.1109/72.701173
7. Gu, D., Hu, H.: Neural predictive control for a car-like mobile robot. Robotics and Autonomous Systems **39**(2), 73–86 (5 2002). https://doi.org/10.1016/S0921-8890(02)00172-0
8. H., V., Leica, P., Roberti, F., Toibero, M., Carelli, R.: Adaptive Coordinated Cooperative Control of Multi-Mobile Manipulators. In: Frontiers in Advanced Control Systems. InTech (2012). https://doi.org/10.5772/39143
9. Hernández-Alvarado, R., García-Valdovinos, L., Salgado-Jiménez, T., Gómez-Espinosa, A., Fonseca-Navarro, F.: Neural Network-Based Self-Tuning PID Control for Underwater Vehicles. Sensors **16**(9), 1429 (9 2016). https://doi.org/10.3390/s16091429, `http://www.mdpi.com/1424-8220/16/9/1429`
10. Kha, N.B., Ahn, K.K.: Position Control of Shape Memory Alloy Actuators by Using Self Tuning Fuzzy PID Controller. In: 2006 1ST IEEE Conference on Industrial Electronics and Applications. pp. 1–5 (5 2006). https://doi.org/10.1109/ICIEA.2006.257198
11. Liu, C., Zhou, C., Cao, W., Li, F., Jia, P.: A Novel Design and Implementation of Autonomous Robotic Car Based on ROS in Indoor Scenario. Robotics **9**(1), 19 (3 2020). https://doi.org/10.3390/robotics9010019, `https://www.mdpi.com/2218-6581/9/1/19`
12. Mannan, M.A., Murata, T., Tamura, J., Tsuchiya, T.: A Fuzzy-Logic-Based Self-Tuning PI Controller for High-Performance Vector Controlled Induction Motor Drive. Electric Power Components and Systems **34**(4), 471–481 (2006). https://doi.org/10.1080/15325000500348194, `https://doi.org/10.1080/15325000500348194`

13. Michel, O.: Cyberbotics Ltd. Webots ¡sup¿™¡/sup¿ : Professional Mobile Robot Simulation. International Journal of Advanced Robotic Systems **1**(1), 5 (3 2004). https://doi.org/10.5772/5618, http://journals.sagepub.com/doi/10.5772/5618
14. Osusky, J., Ciganek, J.: Trajectory tracking robust control for two wheels robot. In: Proceedings of the 29th International Conference on Cybernetics and Informatics, K and I 2018. vol. 2018-January, pp. 1–4. Institute of Electrical and Electronics Engineers Inc. (4 2018). https://doi.org/10.1109/CYBERI.2018.8337559
15. Park, C., Kyung, J.H., Choi, T.Y., Do, H.M., Kim, B.I., Lee, S.H.: Design of an industrial dual arm robot manipulator for a Human-Robot hybrid manufacturing. In: 2012 9th International Conference on Ubiquitous Robots and Ambient Intelligence, URAI 2012. pp. 616–618 (2012). https://doi.org/10.1109/URAI.2012.6463097
16. Recalde, L.F., Guevara, B.S., Cuzco, G., Andaluz, V.H.: Optimal control problem of a differential drive robot. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 12144 LNAI, pp. 75–82. Springer Science and Business Media Deutschland GmbH (9 2020). https://doi.org/10.1007/978-3-030-55789-8_7, https://link.springer.com/chapter/10.1007/978-3-030-55789-8_7
17. Rossomando, F., Soria, C., Patiño, H., Carelli, R.: Model reference adaptive control for mobile robots in trajectory tracking using radial basis function neural networks. Latin American applied research **41**, 177–182 (2011)
18. Varela-Aldas, J., Andaluz, V.H., Chicaiza, F.A.: Modelling and control of a mobile manipulator for trajectory tracking. In: Proceedings - 3rd International Conference on Information Systems and Computer Science, INCISCOS 2018. vol. 2018-Decem, pp. 69–74. Institute of Electrical and Electronics Engineers Inc. (12 2018). https://doi.org/10.1109/INCISCOS.2018.00018
19. Velagic, J., Osmic, N., Lacevic, B.: Design of Neural Network Mobile Robot Motion Controller. In: Ramov, B. (ed.) New Trends in Technologies, chap. 10. IntechOpen, Rijeka (2010). https://doi.org/10.5772/7584, https://doi.org/10.5772/7584
20. Zea, D.J., Guevara, B.S., Recalde, L.F., Andaluz, V.H.: Dynamic Simulation and Kinematic Control for Autonomous Driving in Automobile Robots. pp. 205–216. Springer, Cham (10 2021). https://doi.org/10.1007/978-3-030-63665-4_16, http://link.springer.com/10.1007/978-3-030-63665-4_16