ISSN 2007-9737

Neural Control for a Differential Drive Wheeled Mobile Robot Integrating Stereo Vision Feedback

Michel Lopez-Franco[1], Edgar N. Sanchez[1], Alma Y. Alanis[2], Carlos López-Franco[2]

[1] Instituto Politécnico Nacional, CINVESTAV, Unidad Guadalajara, Jalisco, Mexico

[2] Universidad de Guadalajara, CUCEI, Jalisco, Mexico

{mlopez, sanchez}@gdl.cinvestav.mx, almayalanis@gmail.com, carlos.lopez@cucei.udg.mx

Abstract. This paper proposes a tracking control method for a differential drive wheeled mobile robot with nonholonomic constraints with an inverse optimal neural controller. It is based on two techniques: first, an identifier using a discrete-time recurrent high-order neural network (RHONN) trained with an extended Kalman filter (EKF) algorithm is employed; second, an inverse optimal control is used to avoid solving the Hamilton Jacobi Bellman (HJB) equation. The desired trajectory of the robot is computed during the navigation process using a stereo camera sensor.

Keywords. Neural control, tracking control, differential drive steering, identifier, inverse optimal control.

1 Introduction

One of the greatest achievements in robotics use is in the manufacturing industry. Despite of successes, commercial robots suffer from a fundamental disadvantage, the lack of mobility. A fixed manipulator has a limited range of motion that depends on where it is bolted down. In contrast, a mobile robot would be able to travel throughout the manufacturing plants, flexibly applying its talents wherever it is most effective [1].

Fixed manipulators are typically programmed to perform repetitive tasks with perhaps limited use of sensors, whereas mobile robots are typically less structured in their operation and likely to use more sensors [2].

A mobile robot needs a locomotion mechanism that enables it to move throughout its environment. But there are a large variety of possible ways to move, and so the selection of a robot's approach to locomotion is an important aspect of mobile robot design. In the laboratory, there are research robots that can walk, jump, run, slide, skate, swim, fly, and, of course, roll. Most of these locomotion mechanisms have been inspired from their biological counterparts. There is, however, one exception: the actively powered wheel is a human invention that achieves extremely high efficiency on flat ground [1].

Based on the success of image extraction/interpretation technology and advances in control theory, research has focused on the use of monocular camera-based vision systems for navigating a mobile robot [3-6]. A significant issue with monocular camera-based vision systems is the lack of depth information.

A common type of steering used for mobile robots is differential drive steering illustrated in Fig. 1. Here the wheels on one side of the robot are controlled independently of the wheels on the other side. By coordinating the two different speeds, one can cause the robot to spin in place, move in a straight line, move in a circular path, or follow any prescribed trajectory [7].

The main difficulty of solving the tracking control problem for mobile robots is because the motion of the systems in question has more degrees of freedom than the number of inputs under nonholonomic constraints. As nonholonomic mobile robots have constraints imposed on motions that are not integrable, i.e., the constraints cannot be written as time derivatives of some function of the generalized coordinates, advanced techniques are needed for the tracking control.

A common problem with applying the standard control theory is that the required parameters are often either unknown at design time or are subject to change during operation. For example, the inertia of a robot as seen at the drive motor has many components. These might include the rotational inertia of the motor's rotor, the inertia of gears and shafts, the rotational inertia of its tires, the robot's empty weight, and its payload. Worse yet, there are elements between these components such as bearings, shafts, and belts that may have spring constants and friction loads [8].

1.1 Main Contribution

The objectives of this paper are (1) to propose a controller based on inverse optimal control for mobile robots identified by a RHONN, which includes the robot dynamics and does not require to know the respective parameters; (2) to use visual data for the controller to determine the trajectory references in order to drive the mobile robot from its current pose toward a desired one; (3) to integrate visual servoing and an inverse optimal neural controller allowing mobile robots to perform autonomous navigation.

1.2 Organization of the Paper

The rest of the paper is organized as follows. Section 2 introduces the state model used to express the dynamics of a nonholonomic mobile robot. Section 3 gives an introduction to the inverse control problem. Then, in Section 4 the neural identification problem is described. In Section 5 the neural identification and neural control of mobile robots are presented. In Section 6 the visual feedback for the neural controller is given. The simulation results are presented in Section 7. Finally, conclusions are given in Section 8.

2 Nonholonomic Mobile Robot

In this work we consider a mobile robot with two actuated wheels as shown in Fig. 1. The wheel has been by far the most popular locomotion mechanism in mobile robotics. Wheel robots can be very efficient and balance is not a problem since these robots are designed so that all wheels are in ground contact at all times.

The dynamics of an electrically driven nonholonomic mobile robot can be expressed in the following state-space model [9-11]:

$$\begin{aligned} \dot{\chi}_1 &= J(\chi_1)\chi_2, \\ \dot{\chi}_2 &= M^{-1}(-C(\dot{\chi}_1)\chi_2 - D\chi_2 - \tau_d + NK_T\chi_3), \\ \dot{\chi}_3 &= L_a^{-1}(u - R_a\chi_3 - NK_E\chi_2), \end{aligned} \quad (1)$$

where each subsystem is defined as

$$\begin{aligned} \chi_1 &= [\chi_{11}, \chi_{12}, \chi_{13}]^T, \\ \chi_2 &= [\chi_{21}, \chi_{22}]^T, \\ \chi_3 &= [\chi_{31}, \chi_{32}]^T, \end{aligned}$$

with

$$\begin{aligned} J(\chi_1) &= 0.5r \begin{bmatrix} \cos(\chi_{13}) & \cos(\chi_{13}) \\ \sin(\chi_{13}) & \sin(\chi_{13}) \\ R^{-1} & -R^{-1} \end{bmatrix}, \\ M &= \begin{bmatrix} m_{11} & m_{12} \\ m_{12} & m_{11} \end{bmatrix}, \\ C(\chi) &= 0.5R^{-1}r^2 m_c d \begin{bmatrix} 0 & \dot{\chi}_{13} \\ -\dot{\chi}_{13} & 0 \end{bmatrix}, \\ D &= \begin{bmatrix} d_{11} & 0 \\ 0 & d_{22} \end{bmatrix}, \\ m_{11} &= 0.25R^{-2}r^2(mR^2 + I) + I_w, \\ m_{12} &= 0.25R^{-2}r^2(mR^2 - I), \\ m &= m_c + 2m_w, \\ I &= m_c d^2 + 2m_w R^2 + I_c + 2I_m, \\ \tau &= [\tau_1, \tau_2]^T, \\ \tau_d &= [\tau_{d1}, \tau_{d2}]^T, \end{aligned}$$

where $\chi_{11} = x$, $\chi_{12} = y$ are the coordinates of P_0 and $\chi_{13} = \theta$ is the heading angle of the mobile robot, $\chi_{21} = v_1$, $\chi_{22} = v_2$ represent the angular velocities of right and left wheels, respectively, and $\chi_{31} = i_{a1}$, $\chi_{32} = i_{a2}$ represent motor currents of right and left wheels, respectively. R is half of the width of the mobile robot and r is the radius of the wheel, d is the distance from the center of mass P_c of the mobile robot to the middle point P_0 between the right and left driving wheels, m_c and m_w are the mass of the body and of the wheel with a motor, respectively. I_c, I_w, and I_m are the moments of inertia of the body about the vertical axis through P_c, of the wheel with a motor about the wheel axis, and of the wheel with a motor about the wheel diameter, respectively. The positive terms d_{ii}, $i = 1,2$, are the damping coefficients, $\tau \in \mathbb{R}^2$ is the control torque applied to the wheels of the robot, $\tau_d \in \mathbb{R}^2$ is a vector of disturbances including

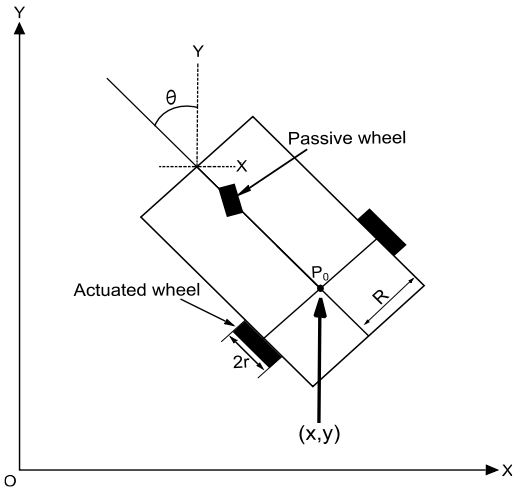

Fig. 1. Mobile robot with two actuated wheels

unmodeled dynamics. $K_T = diag[k_{t_1}, k_{t_2}]$ is the motor torque constant, i_a =$[i_{a_1}, i_{a_2}]$ is the motor current vector, $u \in \mathbb{R}^2$ is the input voltage, $R_a = diag[r_{a_1}, r_{a_2}]$ is the resistance, $L_a = diag[l_{a_1}, l_{a_2}]$ is the inductance, $K_E = diag[k_{e_1}, k_{e_2}]$ is the back electromotive force coefficient, and $N = diag[n_1, n_2]$ is the gear ratio. Here, $diag[\cdot]$ denotes the diagonal matrix. Model (1) is discretized using the Euler Methodology.

3 Inverse Optimal Control

The main goal of this section is a synthesis of an inverse optimal control. First, we briefly give details about optimal control methodology and their limitations. Let us consider a discrete-time affine-in-the-input nonlinear system

$$\chi_{k+1} = f(\chi_k) + g(\chi_k)u_k,\ \chi_0 = \chi(0), \tag{2}$$

where $\chi_k \in \mathbb{R}^n$ is the state of the system at time $k \in \mathbb{N}$, $u \in \mathbb{R}^m$ $f: \mathbb{R}^n \to \mathbb{R}^n$, $g: \mathbb{R}^n \to \mathbb{R}^{n\times m}$ are smooth and bounded mappings. We assume $f(0) = 0$. $\mathbb{N}$ denotes the set of nonnegative integers.

The following meaningful cost functional is associated with the trajectory tracking problem for system (2):

$$\mathcal{L}(z_k) = \sum_{n=k}^{\infty} (l(z_n) + u_n^T R(z_n) u_n). \tag{3}$$

where $z_k = \chi_k - \chi_{\delta,k}$ with $\chi_{\delta,k}$ as the desired trajectory for χ_k; $z_k \in \mathbb{R}^n$; $\mathcal{L}(z_k): \mathbb{R}^n \to \mathbb{R}^+$; $l(z_k): \mathbb{R}^n \to \mathbb{R}^+$ is a positive semi-definite function and $R(z_k): \mathbb{R}^n \to \mathbb{R}^m \times m$ is a real symmetric positive definite weighting matrix. The entries of $R(z_k)$ can be fixed or can be functions of the system state in order to vary the weighting on control efforts according to the state value [12].

Considering the state feedback control design problem, we assume that the full state χ_k is available. Using the optimal value function $\mathcal{L}^*(z_k)$ for (3) as Lyapunov function $V(z_k)$, equation (3) can be rewritten as

$$\begin{aligned} V(z_k) &= l(z_k) + u_k^T R(z_k) u_k \\ &\quad + \sum_{n=k+1}^{\infty} (l(z_n) + u_n^T R(z_n) u_n) \\ &= l(z_k) + u_k^T R(z_k) u_k + V(z_{k+1}), \end{aligned}$$

where the boundary condition $V(0) = 0$ is required so that $V(z_k)$ becomes a Lyapunov function.

From the Bellman optimality principle [13, 14] it is known that, for the infinite horizon optimization case, the value function $V(z_k)$ becomes time invariant and satisfies the discrete-time (DT) Bellman equation [13, 15, 16]

$$V(z_k) = \min_{u_k} \{l(z_k) + u_k^T R(z_k) u_k + V(z_{k+1})\},$$

where $V(z_{k+1})$ depends on both z_k and u_k by means of z_{k+1} in (2). Note that the DT Bellman equation is solved backward in time [15]. In order to establish the conditions that the optimal control law must satisfy, we define the discrete-time Hamiltonian $H(z_k, u_k)$ as

$$\begin{aligned} H(z_k, u_k) = l(z_k) + u_k^T R(z_k) u_k + V(z_{k+1}) \\ - V(z_k). \end{aligned} \tag{4}$$

A necessary condition that the optimal control law should satisfy is $\frac{\partial H(z_k, u_k)}{\partial u_k} = 0$, then

$$\begin{aligned} 0 &= 2R(z_k)u_k + \frac{\partial V(z_{k+1})}{\partial u_k} \\ &= 2R(z_k)u_k + \frac{\partial z_{k+1}}{\partial u_k}\frac{\partial V(z_{k+1})}{\partial z_{k+1}} \\ &= 2R(z_k)u_k + g^T(\chi_k)\frac{\partial V(z_{k+1})}{\partial z_{k+1}}. \end{aligned}$$

Therefore, the optimal control law to achieve trajectory tracking is formulated as

$$u_k^* = -\frac{1}{2}R^{-1}(z_k)g^T(\chi_k)\frac{\partial V(z_{k+1})}{\partial z_{k+1}},$$

with the boundary condition $V(0) = 0$. For solving the trajectory tracking inverse optimal control problem, it is necessary to solve the following HJB equation:

$$l(z_k) + V(z_{k+1}) - V(z_k) + \frac{1}{4}\frac{\partial V^T(z_{k+1})}{\partial z_{k+1}}g(\chi_k). \qquad (5)$$

Solving the HJB partial differential equation (5) is not straightforward; this is one of the main disadvantages of discrete-time optimal control for nonlinear systems. To overcome this problem, we propose to solve the inverse optimal control problem.

Definition 1. Consider the tracking error as $z_k = \chi_k - \chi_{\delta,k}$, with $\chi_{\delta,k}$ being the desired trajectory for χ_k. Let us define the control law as

$$u_k^* = -\frac{1}{2}R^{-1}(z_k)g^T(\chi_k)\frac{\partial V(z_{k+1})}{\partial z_{k+1}}, \qquad (6)$$

which will be inverse optimal (globally) stabilizing along the desired trajectory $\chi_{\delta,k}$ if

- **(i)** it achieves (global) asymptotic stability of $\chi_k = 0$ for system (2) along reference $\chi_{\delta,k}$;
- **(ii)** $V(z_k)$ is a (radially unbounded) positive definite function such that inequality

  $$\overline{V} := V(z_{k+1}) - V(z_k) + u_k^{*T}R(z_k)u_k^* \leq 0$$

  is satisfied.

Selecting $l(z_k) := -\overline{V}$, then $V(z_k)$ is a solution for (5) and cost functional (3) is minimized.

As established in *Definition* 1, the inverse optimal control law for trajectory tracking is based in knowledge of $V(z_k)$.Then, a CLF $V(z_k)$ is proposed such that (i) and (ii) are guaranteed. Hence, instead of solving (5) for $V(z_k)$, a quadratic candidate CLF $V(z_k)$ for (6) is proposed with the form

$$V(z_k) = \frac{1}{2}z_k^T P z_k \; P = P^T > 0 \qquad (7)$$

in order to ensure stability of the tracking error z_k, where

$$z_k = \chi_k - \chi_{\delta,k} = \begin{bmatrix} \chi_{1,k} - \chi_{1\delta,k} \\ \vdots \\ \chi_{n,k} - \chi_{n\delta,k} \end{bmatrix}.$$

The control law (6) with (7), which is referred to as the inverse optimal control law, optimizes the meaningful cost functional of the form in (3). Consequently, by considering $V(z_k)$ as in (7), the control law in (6) takes the following form:

$$\begin{aligned} u_k^* &= -\frac{1}{4}R^{-1}(z_k)g^T(\chi_k)\frac{\partial z_{k+1}^T P z_{k+1}}{\partial z_{k+1}} \\ &= -\frac{1}{4}R^{-1}(z_k)g^T(\chi_k)Pz_{k+1} \\ &= -\frac{1}{2}(R(z_k)g^T(\chi_k)Pg(z_k))^{-1} \\ &\quad \times g^T(\chi_k)P\big(f(\chi_k) - \chi_{\delta,k+1}\big). \end{aligned} \qquad (8)$$

It is worth pointing out that P and $R(z_k)$ are positive definite and symmetric matrices; thus, the existence of the inverse in (6) is ensured.

4 Neural Identification

To identify the system in (2), we use a discrete-time recurrent high-order neural network (RHONN) defined as

$$x_{i,k+1} = w_i^{\top}\varphi_i(\chi_k, u_k), \;\; i = 1, \cdots, n, \qquad (9)$$

where x_i is the state of the i-th neuron, L_i is the respective number of high-order connections, $\{I_1, I_2, \cdots, I_{L_i}\}$ is a collection of non-ordered subsets of $\{1,2, \cdots, n+m\}$, n is the state dimension, m is the number of external inputs, w_i is the respective on-line adapted weight vector, and $\varphi_i(x_k, u_k)$ is given by

$$\varphi_i(x_k,u_k)=\begin{bmatrix}\varphi_{i_1}\\ \varphi_{i_2}\\ \vdots\\ \varphi_{i_{L_i}}\end{bmatrix}=\begin{bmatrix}\prod_{j\in I_1}\xi_{i_j}^{d_{i_j}(1)}\\ \prod_{j\in I_2}\xi_{i_j}^{d_{i_j}(2)}\\ \vdots\\ \prod_{j\in I_{L_i}}\xi_{i_j}^{d_{i_j}(L_i)}\end{bmatrix}, \tag{10}$$

with $d_{i_j}(k)$ being nonnegative integers and ξ_i defined as follows:

$$\xi_i=\begin{bmatrix}\xi_{i_1}\\ \vdots\\ \xi_{i_1}\\ \xi_{i_{n+1}}\\ \vdots\\ \xi_{i_{n+m}}\end{bmatrix}=\begin{bmatrix}S(x_1)\\ \vdots\\ S(x_n)\\ u_1\\ \vdots\\ u_m\end{bmatrix}, \tag{11}$$

where $u=[u_1,u_2,\dots,u_m]^\top$ is the input vector to the neural network and $S(\bullet)$ is defined by

$$S(\Upsilon)=\frac{1}{1+\exp(-\beta\Upsilon)},\ \beta>0, \tag{12}$$

where Υ is any real value variable.

Using the structure of system in (2), we propose the following discrete-time RHONN series-parallel representation [17]:

$$x_{i,k+1}=w_i^{*\top}\varphi_i(x_k,u_k)+\epsilon_{\varphi_i},\ \ i=1,\cdots,n, \tag{13}$$

where ϵ_{z_i} is a bounded approximation error which can be reduced by increasing the number of the adjustable weights [17].

Assume that there exists an ideal weight vector w_i^* such that $\|\epsilon_{z_i}\|$ can be minimized on a compact set $\Omega_{z_i}\subset\Re^{L_i}$ The ideal weight vector w_i^* is an artificial quantity required for analytical purpose [17]. In general, it is assumed that this vector exists and is constant but unknown. Let us define its estimate as w_i and the estimation error as

$$\tilde{w}_{i,k}=w_{i,k}-w_i^*. \tag{14}$$

The RHONN is trained with an Extended Kalman Filter (EKF) algorithm in (17). Then, the dynamics of the identification error in (19) can be expressed as

$$e_{i,k+1}=\tilde{w}_{i,k}\varphi_i(\chi_k,u_k)+\epsilon_{z_i}. \tag{15}$$

On the other hand, the dynamics of (14) is

$$\tilde{w}_{i,k+1}=\tilde{w}_{i,k}-\eta_i K_{i,k}e_k. \tag{16}$$

It is possible to identify (2) by (9) due to the theorem that follows.

Theorem 1 [18]. The RHONN in (9) trained with the EKF-based algorithm in (17) to identify the non-linear plant in (2) ensures that the identification error in (19) is semiglobally uniformly ultimately bounded (SGUUB); moreover, the RHONN weights remain bounded.

4.1 The EKF Training Algorithm

The best well-known training approach for recurrent neural networks (RNN) is the back propagation through time learning [19]. However, it is a first order gradient descent method and hence its learning speed can be very slow [20]. Recently, Extended Kalman Filter (EKF) based algorithms have been introduced to train neural networks [21, 22]. With an EKF based algorithm, the learning convergence is improved [20]. The EKF training of neural networks, both feedforward and recurrent ones, has proven to be reliable and practical for many applications over the past ten years [22].

It is known that Kalman filtering (KF) estimates the state of a linear system with additive state and output white noises [23, 24]. For KF-based neural network training, the network weights become the states to be estimated. In this case, the error between the neural network output and the measured plant output can be considered as additive white noise. Due to the fact that the neural network mapping is nonlinear, an EKF-type is required (see [22] and references therein).

The training goal is to find the optimal weight values which minimize the prediction error. The EKF-based training algorithm is described in [23] as

$$\begin{aligned}K_{i,k} &= P_{i,k}H_{i,k}M_{i,k},\\ w_{i,k+1} &= w_{i,k}+\eta_i K_{i,k}e_{i,k},\\ P_{i,k+1} &= P_{i,k}-K_{i,k}H_{i,k}^\top P_{i,k}+Q_{i,k},\end{aligned} \tag{17}$$

with

$$M_{i,k} = \left[R_{i,k}+H_{i,k}^\top P_{i,k}H_{i,k}\right]^{-1}, \tag{18}$$

$$e_{i,k} = \chi_{i,k}-x_{i,k}, \tag{19}$$

where $e_{i,k}$ is the identification error, $P_{i,k} \in \Re^{L_i \times L_i}$ is the state estimation prediction error covariance matrix, $w_i \in \Re^{L_i}$ is the weight (state) vector, L_i is the total number of neural network weights, $\chi_i \in \Re$ is the i-th plant state component, $x_i \in \Re$ is the i-th neural state component, η_i is a design parameter, $K_i \in \Re^{L_i \times m}$ is the Kalman gain matrix, $Q_i \in \Re^{L_i \times L_i}$ is the state noise associated covariance matrix, $R_i \in \Re^{m \times m}$ is the measurement noise associated covariance matrix, $H_i \in \Re^{L_i \times m}$ is a matrix for which each entry (H_{ij}) is the derivative of one of the neural network output (x_i) with respect to one neural network weight (w_{ij}) as follows:

$$H_{ij,k} = \left[\frac{\partial x_{i,k}}{\partial w_{ij,k}}\right]_{w_{i,k}=\hat{w}_{i,k+1}}, \quad i = 1,\ldots,n \text{ and } j = 1,\ldots,L_i. \tag{20}$$

Usually, P_i, Q_i, and R_i are initialized as diagonal matrices, with entries $P_i(0)$, $Q_i(0)$, and $R_i(0)$, respectively. It is important to note that $H_{i,k}$, $K_{i,k}$, and $P_{i,k}$ for the EKF are bounded [24].

Proposition 1. The tracking of a desired trajectory x_δ defined in terms of the plant state χ formulated as in (2) can be established as the following inequality [26]:

$$\| x_\delta - \chi \| \leq \| x - \chi \| + \| x_\delta - x \|, \tag{21}$$

where $\|\bullet\|$ stands for the Euclidean norm, $x_\delta - \chi$ is the system output tracking error; $x - \chi$ is the output identification error; and $x_\delta - x$ is the RHONN output tracking error.

We establish the requirements for the tracking solution as follows.

Requirement 1.

$$\lim_{t\to\infty} \| x - \chi \| \leq \zeta \tag{22}$$

with ζ being a small positive constant.

Requirement 2.

$$\lim_{t\to\infty} \| x_\delta - x \| = 0. \tag{23}$$

An on-line neural identifier based on (13) ensures (22), while (23) is guaranteed by a discrete-time inverse optimal control. It is possible to establish Proposition 1 due to the separation principle for discrete-time nonlinear systems [27].

5 Neural Identification and Control of Nonholonomic Mobile Robots

In this section we describe the neural identification and the neural control of a nonholonomic mobile robot.

5.1 Neural Identification Design

The physical parameters for the mobile robot simulations are selected as

$$\begin{array}{cc} R = 0.75m & I_m = 0.0025kgm^2 \\ d = 0.3m & R_a = diag[2.5,2.5]\Omega \\ r = 0.15m & L_a = diag[0.048,0.048]H \\ m_c = 30kg & K_E = diag[0.02,0.02]V/\frac{rad}{s} \\ m_w = 1kg & N = diag[62.55,62.55] \\ I_c = 15.625kgm^2 & K_T = diag[0.2613,0.2613]\frac{Nm}{A} \\ I_w = 0.005kgm^2 & d_{m1} = d_{m2} = 0.5N. \end{array}$$

To this end, we apply the neural identifier, developed in Section 4, to obtain a discrete-time neural model for the electrically driven nonholonomic mobile robot in (1), with $n = 7$ trained with the EKF in (17) as follows:

$$\begin{aligned} x_{1,k+1} &= w_{11,k}S(\chi_{11,k}) + w_{12,k}S(\chi_{12,k}) + w'_{1_{11}}\chi_3 + w'_{1_{12}}\chi_4, \\ x_{2,k+1} &= w_{21,k}S(\chi_{11,k}) + w_{22,k}S(\chi_{12,k}) + w'_{1_{21}}\chi_3 + w'_{1_{22}}\chi_4, \\ x_{3,k+1} &= w_{31,k}S(\chi_{11,k}) + w_{32,k}S(\chi_{12,k}) + w'_{1_{31}}\chi_3 + w'_{1_{32}}\chi_4, \\ x_{4,k+1} &= w_{41,k}S(\chi_{11,k}) + w_{42,k}S(\chi_{12,k}) + w_{43,k}S(\chi_{21,k}) + w_{44,k}S(\chi_{31,k}) + w'_2\chi_6, \\ x_{5,k+1} &= w_{51,k}S(\chi_{11,k}) + w_{52,k}S(\chi_{12,k}) + w_{53,k}S(\chi_{22,k}) + w_{54,k}S(\chi_{32,k}) + w'_2\chi_7, \\ x_{6,k+1} &= w_{61,k}S(\chi_{11,k}) + w_{62,k}S(\chi_{12,k}) + w_{63,k}S(\chi_{21,k}) + w_{64,k}S(\chi_{31,k}) + w'_3u_{11}, \\ x_{7,k+1} &= w_{71,k}S(\chi_{11,k}) + w_{72,k}S(\chi_{12,k}) + w_{73,k}S(\chi_{22,k}) + w_{74,k}S(\chi_{32,k}) + w'_3u_{12}. \end{aligned} \tag{24}$$

where x_1 and x_2 identify the x and y coordinates, respectively; x_3 identifies the robot angle; x_4 and

x_5 identify the angular velocities of the right and left wheels, respectively; finally, x_6 and x_7 identify the motor currents, respectively. The NN training is performed on-line, and all of its states are initialized in a random way. The RHONN parameters are heuristically selected as

$$\begin{array}{lll} P_{1,(0)} = 1 \cdot 10^8 & R_{1,(0)} = 1 \cdot 10^4 & Q_{1,(0)} = 5 \cdot 10^5 \\ P_{2,(0)} = 1 \cdot 10^2 & R_{2,(0)} = 5 \cdot 10^4 & Q_{2,(0)} = 5 \cdot 10^5 \\ P_{3,(0)} = 1 \cdot 10^8 & R_{3,(0)} = 1 \cdot 10^4 & Q_{3,(0)} = 5 \cdot 10^5 \\ P_{4,(0)} = 1 \cdot 10^2 & R_{4,(0)} = 1 \cdot 10^1 & Q_{4,(0)} = 1 \cdot 10^1 \\ P_{5,(0)} = 1 \cdot 10^2 & R_{5,(0)} = 1 \cdot 10^1 & Q_{5,(0)} = 1 \cdot 10^1 \\ P_{6,(0)} = 1 \cdot 10^2 & R_{6,(0)} = 1 \cdot 10^3 & Q_{6,(0)} = 1 \cdot 10^3 \\ P_{7,(0)} = 1 \cdot 10^2 & R_{7,(0)} = 1 \cdot 10^3 & Q_{7,(0)} = 1 \cdot 10^3. \end{array}$$

It is important to consider that for the EKF-learning algorithm the covariances are used as design parameters [22, 28]. The neural network structure in (24) is determined heuristically in order to minimize the state estimation error. The results are presented in what follows.

5.2 Control Synthesis

In order to facilitate the controller synthesis, we rewrite the neural network in (24) in a block structure form as

$$\begin{array}{rcl} x_{1,k+1} = \begin{bmatrix} x_{1,k+1} \\ x_{2,k+1} \\ x_{3,k+1} \end{bmatrix} & = & w_{1,k}\varphi_1(\chi_{1,k}) + w'_{1,k}\chi_{2,k}, \\ x_{2,k+1} = \begin{bmatrix} x_{4,k+1} \\ x_{5,k+1} \end{bmatrix} & = & w_{2,k}\varphi_2(\chi_{1,k}, \chi_{2,k}) \\ & & + w'_{2,k}\chi_{3,k}, \\ x_{3,k+1} = \begin{bmatrix} x_{6,k+1} \\ x_{7,k+1} \end{bmatrix} & = & w_{3,k}\varphi_3(\chi_{1,k}, \chi_{2,k}, \chi_{3,k}) \\ & & + w'_{3,k}u_k. \end{array} \quad (25)$$

with $\chi_{1,k}$, $\chi_{2,k}$, $\chi_{3,k}$, φ_1, φ_2, φ_3, $w_{1,k}$, $w_{2,k}$, $w_{3,k}$, $w'_{1,k}$, $w'_{2,k}$, and $w'_{3,k}$ of appropriated dimension according to (25).

For trajectory tracking of the first block in (25), let us define the tracking error as

$$z_{1,k} = x_{1,k} - \chi_{1\delta,k},$$

where $\chi_{1\delta,k}$ is the desired trajectory. Then using (25) and introducing the desired dynamics for $z_{1,k}$ result in

$$\begin{array}{rcl} z_{1,k+1} & = & w_{1,k}\varphi_1(\chi_{1,k}) + w'_{1,k}\chi_{2,k} - \chi_{1\delta,k+1} \\ & = & w_{1,k}\varphi_1(z_{1,k}) + K_1 z_{2,k}, \end{array} \quad (26)$$

where $K_1 = diag\{k_{1_1}, k_{2_1}, k_{3_1}\}$ with $|k_{1_1}|, |k_{2_1}|, |k_{3_1}| < 1$. The desired value $\chi_{2\delta,k}$ for the pseudo-control input $\chi_{2,k}$ is calculated from (26) as

$$\begin{array}{rcl} \chi_{2\delta,k} & = & (w'_{1,k})^{-1}(-w_{1,k}\varphi_1(\chi_{1,k}) + \chi_{1\delta,k+1} \\ & & w_{1,k}\varphi_1(z_{1,k}) + K_1 z_{1,k}). \end{array} \quad (27)$$

Note that the calculated value of the state $\chi_{2\delta,k}$ in (27) is not the true value of such state; instead, it represents the desired behavior for $x_{k,2}$. To avoid misunderstandings, the desired value for $x_{2,k}$ is referred to as $x_{2\delta,k}$ in (27):

$$z\_2 = x\_(2,k) - \chi\_(2\delta,k).$$

Then using (25) and introducing the desired dynamics for $z_{2,k}$ result in

$$\begin{array}{rcl} z_{2,k+1} & = & w_{2,k}\varphi_2(\chi_{1,k}, \chi_{2,k}) + w'_{2,k}\chi_{3,k} - \chi_{2\delta,k+1} \\ & = & w_{2,k}\varphi_2(z_{1,k}, z_{2,k}) + K_2 z_{3,k}, \end{array} \quad (28)$$

where $K_2 = diag\{k_{1_2}, k_{2_2}\}$ with $|k_{1_2}|, |k_{2_2}| < 1$. The desired value $\chi_{3\delta,k}$ for the pseudo-control input $\chi_{3,k}$ is calculated from (28) as

$$\begin{array}{rcl} \chi_{3\delta,k} & = & (w'_{2,k})^{-1}(-w_{2,k}\varphi_2(\chi_{1,k}, \chi_{2,k}) + \chi_{2\delta,k+1} \\ & & w_{2,k}\varphi_2(z_{1,k}, z_{2,k}) + K_2 z_{2,k}). \end{array} \quad (29)$$

At the third step, we introduce a new variable as $z_3 = x_{3,k} - \chi_{3\delta,k}$.

Taking one step ahead, we have

$$z_{3,k+1} = w_{3,k}\varphi_3(\chi_{1,k}, \chi_{2,k}, \chi_{3,k}) + w'_{3,k}u_k - \chi_{3\delta,k+1} \quad (30)$$

$$= w_{3,k}\varphi_2(z_{1,k}, z_{2,k}, z_{3,k}) + K_3 u_k. \quad (31)$$

Then, the system in (25) can be presented with the new variables $z = [z_1^T, z_2^T z_3^T]^T$ as

$$\begin{array}{rcl} z_{1,k+1} & = & w_{1,k}\varphi_2(z_{1,k}) + K_1 z_{1,k} \\ z_{2,k+1} & = & w_{2,k}\varphi_2(z_{1,k}, z_{2,k}) + K_2 z_{2,k} \\ z_{3,k+1} & = & w_{3,k}\varphi_2(z_{1,k}, z_{2,k}, z_{3,k}) + K_3 u_k, \end{array} \quad (32)$$

where u_k is defined as

$$\begin{array}{rcl} u_k & = & -\frac{1}{2}(R(z_k) + g^T(x_k)Pg(z_k))^{-1} \\ & & \times g^T(x_k)P(f(z_k)). \end{array} \quad (33)$$

where the controllers parameters are selected heuristically as

$$P = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

and

$$w'_{1,k} = \begin{bmatrix} \cos(x_3) & \cos(x_3) \\ \sin(x_3) & \sin(x_3) \\ R^{-1} & R^{-1} \end{bmatrix},$$

$$w'_{2,k} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \text{ and } w'_{3,k} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

6 Visual Feedback

The use of visual feedback to control a robot is commonly termed as visual servoing or visual control [29, 30]. In this work the visual data is acquired from a stereo vision system that is mounted directly on the mobile robot, see Fig. 2.

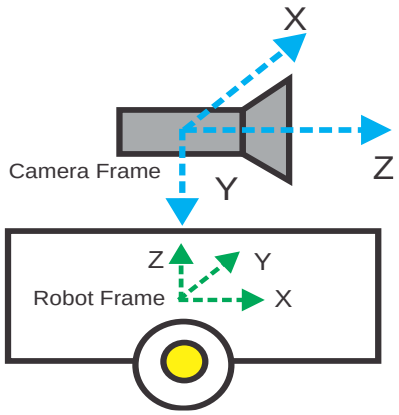

Fig. 2. Coordinate systems of the mobile robot and the stereo vision system

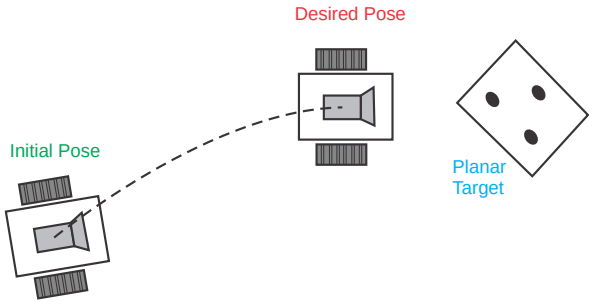

Fig. 3. Robot moving from the initial pose to the desired pose

When the camera is mounted on the robot, its motion induces camera motion, see Fig. 3.

The visual control objective is to minimize an error $e(t)$ defined as [31]

$$e(t) = s(t) - s^*, \tag{34}$$

where $s(t)$ denotes the features extracted from the current pose and s^* denotes the features extracted from the desired pose.

In this paper we consider a nonholonomic mobile robot moving on a plane as shown in Fig. 1. Its pose is defined as $[x\ y\ \theta]^T$. Its kinematics model is that of a wheeled unicycle mobile robot:

$$\begin{aligned} \dot{x} &= v_r \cos\theta, \\ \dot{y} &= v_r \sin\theta, \\ \dot{\theta} &= \omega_r, \end{aligned} \tag{35}$$

where v_r and ω_r represent the translational and angular velocities, respectively.

In order to estimate v_r and ω_r by using visual data, several steps must be made, Fig. 4. First, the image is converted to HSV (Hue Saturation Value) color space [32]. Using this image we apply a mask, previously computed from a reference image, and then we obtain a segmented image. From the segmented image, we compute the boundaries using the Moore-neighbor tracing algorithm [33]; then to each boundary we compute the metric $m = 4\pi\ area/perimeter^2$, if this is close to 1 then the boundary is more likely to be a circle.

From the detected circles, we compute their centroid. Later, using the centroids of the circles from the desired image, the current image, and the corresponding depths, we estimate the robot's pose. Finally, with the current and desired poses, we compute the velocities v_r, ω_r to drive the robot from the current pose to the desired pose.

6.1 Stereo Vision

The principle of stereo vision with parallel optical axes is displayed in Fig. 5. The 3D point P is projected onto the image plane of the left camera as $p_L = [x_l, y_l]$, similarly $p_R = [x_R, y_R]$ represents the projection of P onto the image plane of the right camera.

Since the image planes of the left and right cameras are located on the same plane, the y-coordinates in these two images are the same ($y_L = y_R$), and the disparity is equal to the

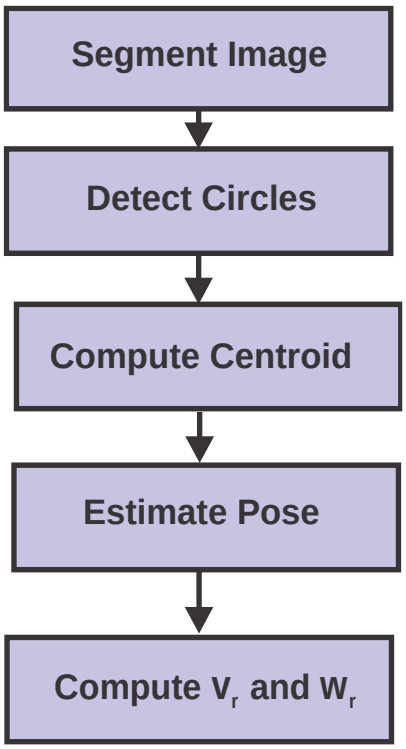

Fig. 4. Process for the computation of the translational v_r, ω_r and angular velocities from visual data

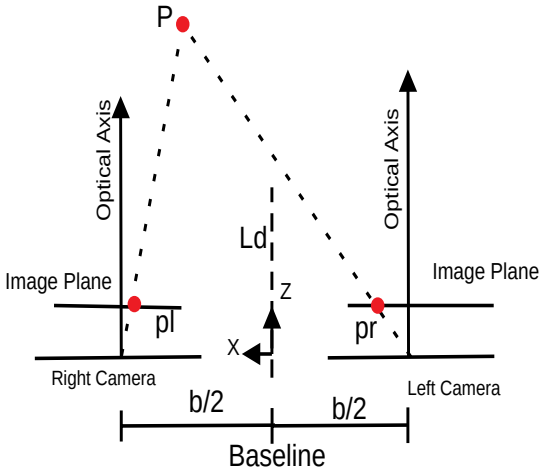

Fig. 5. Image formation of stereo vision with parallel optical axes

difference between the horizontal coordinates $(x_R - x_L)$.

Let $P = (X, Y, Z) \in \mathbb{R}^3$ denote a 3D point in the world. The coordinates of P on the left camera are

$$P_L = [X + b/2, Y, Z]^T. \quad (36)$$

Similarly, the point P on the right camera is

$$P_R = [X - b/2, Y, Z]^T. \quad (37)$$

Using the standard projective camera projection, we obtain

$$x_L = \frac{\left(X+\frac{b}{2}\right)f}{Z}, \quad (38)$$

$$x_R = \frac{\left(X-\frac{b}{2}\right)f}{Z}, \quad (39)$$

Similarly,

$$y_L = \frac{Yf}{Z}, \quad (40)$$

$$y_R = \frac{Yf}{Z}. \quad (41)$$

The depth of the point P can be recovered from the x-coordinate of the image points x_L and x_R, subtracting (39) from (38) we obtain

$$Z = \frac{bf}{x_L - x_R} \quad (42)$$

Similarly, we can also solve for X using (38), (39), and (81) and obtain

$$X = \frac{b(x_L + x_R)}{2(x_L - x_R)}. \quad (43)$$

The Y value can be recovered with (40) or (41), since they have the same value, and from (42) to get

$$Y = \frac{by}{x_L - x_R}. \quad (44)$$

6.2 Pose Estimation

The mobile robot moves on a 2D plane, thus we need only two coordinates to fully determine its pose (x,y,θ). Since the robot cannot move in the Y direction (orthogonal to the plane), we can estimate its pose with respect to the planar target using only the Z and X values of the point P.

Let $Q_i^* = (Z_i^*, X_i^*)$ and $Q_i = (Z_i, X_i)$ represent a 2D Euclidean point of a feature point P_i expressed in the frames $\mathcal{F}^*$ and $\mathcal{F}$, respectively. From Euclidean geometry, the relationship between the features is defined as

$$Q_i^* = RQ_i + t, \tag{45}$$

where $R \in \mathbb{R}^{2\times2}$ is the 2D rotation matrix and $t = (t_x, t_y) \in \mathbb{R}^2$ is the translation vector, Fig. 6.

To estimate the pose of the robot given the points Q_i and Q_i^* from the current and desired pose, we need to solve the following least-square problem:

$$E(\theta, t) = \sum_{i=1}^{n} | R_\theta X_i + t - X_i^* |^2. \tag{46}$$

This problem can be solved in a closed form [34].

6.3 Kinematic Planer

Once the pose of the robot has been estimated, the next step is the estimation of the robot velocities which minimize the error between the current pose of the robot and its desired pose.

The path to track is defined as the line L_d passing through the center of the stereo rig parallel to the optical axes of the left and right cameras, Fig. 6. A line on the plane can be defined using the general equation of the line $(ax + by + c = 0)$, therefore, the desired line is defined at the desired pose as

$$L_d = [0\ 1\ 0]. \tag{47}$$

The signed distance from the current pose of the robot and the principal axis L_d at the desired pose is defined as

$$d = [t_x\ t_y\ 1]^t \cdot L_d. \tag{48}$$

The angular velocity of the robot must turn the robot toward the line L_d with

$$\beta_d = -K_d d, \quad K_d > 0, \tag{49}$$

and adjust the orientation of the robot (heading angle) with

$$\beta_o = K_o(\theta^* - \theta), \quad K_o > 0. \tag{50}$$

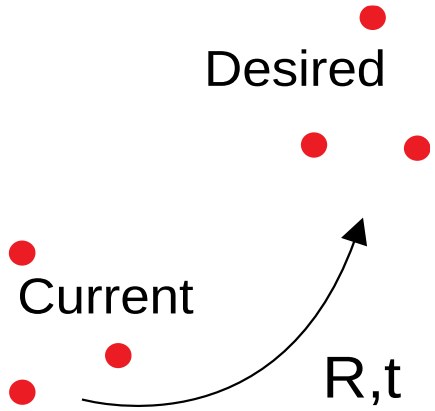

Fig. 6. Pose estimation problem

Then, the combined kinematic control law [8] used to generate the robot's angular velocity for path following is defined as

$$\omega_r = \beta_d + \beta_o. \tag{51}$$

The value of v_r is set to a constant value (e.g., 0.2 m/s), but when the robot is close to the desired pose, the velocity is computed with

$$v_r = \kappa_v \sqrt{t_x^2 + t_y^2}. \tag{52}$$

7 Simulation Results

In this section we present the simulation results of our proposed discrete-time inverse optimal neural controller with stereo vision feedback. Simulations have been performed using Matlab-Simulink.

In the simulation, the robot moves under the action of the proposed controller, the controller uses as references the linear and angular velocities computed from the stereo vision algorithm. In the simulation, the initial pose of the robot is $[0\ 0\ 0]^T$, and the desired pose is $[3.8 - 0.8\ 0]^T$.

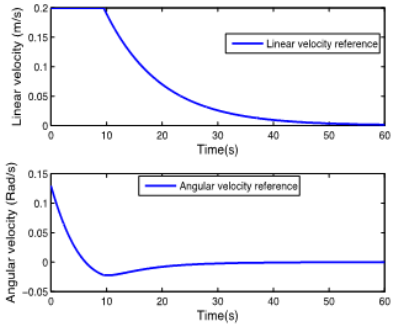

Fig. 7. Linear (top) and angular (bottom) velocities generated by the stereo vision algorithm

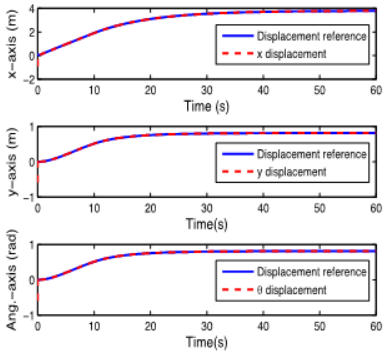

Fig. 8. x-axis tracking (top), y-axis tracking (middle), and θ tracking (bottom), reference signal in solid line and plant signal in dashed line

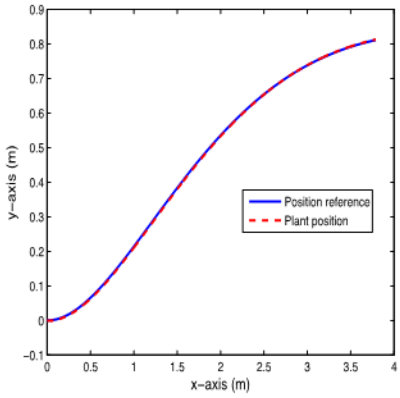

Fig. 9. Trajectory tracking result for simulation (reference signal in solid line and plant signal in dashed line)

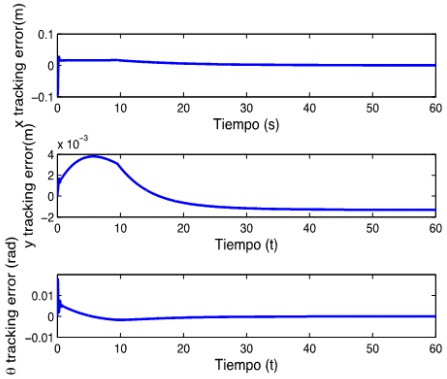

Fig. 10. Tracking errors, x-axis (top), y-axis (middle), and θ angle (bottom)

The sampling time of the simulation was T= 0.01s. The simulation results are presented as follows. In Fig. 7 we show the linear and angular velocities used as references by the proposed controller, these velocities are computed by the stereo vision algorithm using the current and desired images of the target object.

Fig. 8 shows the tracking performance for x-axis, y-axis, and θ angle.

Fig. 9 shows the trajectory tracking results.

Fig. 10 presents the tracking errors.

Fig. 11 shows the applied control signal for the left and right wheels.

Fig. 12 presents the current tracking for the left and right wheels.

Fig. 13 shows the angular velocity tracking for the left and right wheels, respectively.

Finally, Fig. 14 portraits the robot moving from the initial pose to the desired pose.

In order to compare the proposed control scheme with the works already published [36], Table 1 is included, which is described as follows: the controllers used in this comparison are (1) Neural Backstepping Controller (NBC) [36], (2) High-Order Sliding Mode Controller (HOSM) [36],

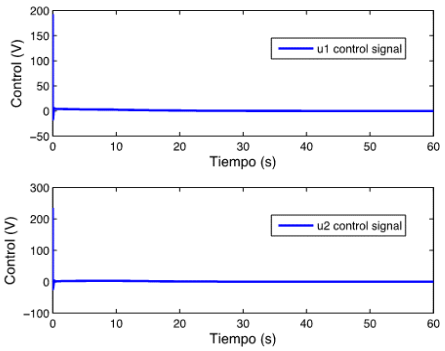

Fig. 11. Applied control signal for the left and right wheels, respectively

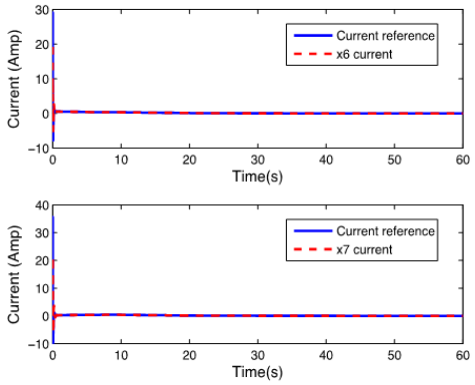

Fig. 12. Current tracking for simulation in the left and right wheels, respectively (tracking signal in solid line and plant signal in dashed line)

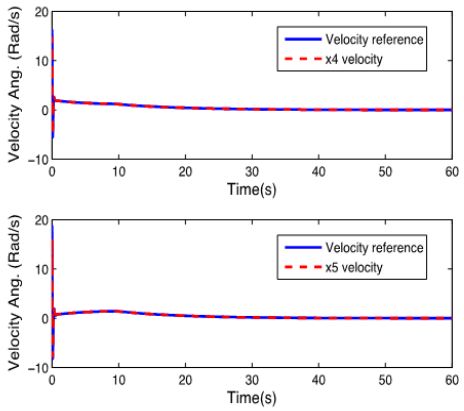

Fig. 13. Angular velocity tracking for simulation in left and right wheels, respectively (tracking signal in solid line and plant signal in dashed line)

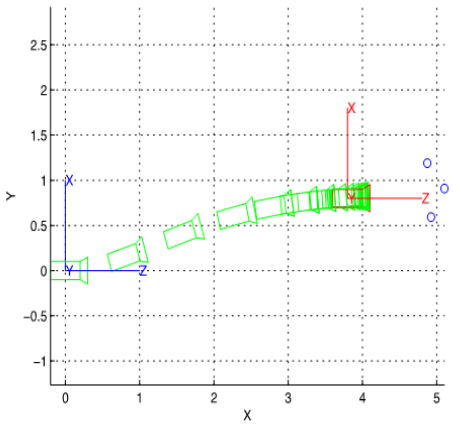

Fig. 14. Robot moving from the initial pose to the desired pose

and (3) Inverse Optimal Neural Controller (IONC) proposed in this paper.

Table 1. Comparison between Inverse Optimal Neural Controller (IONC) with respect to Neural Backstepping Controller (NBC) and High-Order Sliding Mode Controller (HOSM)

Method	Mean Value	Standard Deviation
NBC	-0.0149	0.0894
HOSM	-0.0223	0.0967
IONC	0.0034	0.0398

8 Conclusions

In this work we presented a neural identification and a neural controller for a nonholonomic mobile robot. The proposed controller allows the robot to accomplish a trajectory tracking problem of a nonlinear system.

The controller is inverse optimal in the sense that it minimizes a meaningful cost functional. The mobile robot dynamics at the actuator level as well as its kinematics and dynamics uncertainties are considered in the construction of the controller by

means of neural identification. The references for the controller are provided by a visual sensor.

The obtained results show the effectiveness of the proposed controller.

Acknowledgements

The authors thank CONACYT, Mexico, for the support through Projects 103191Y, 156567Y, and INFR- 229696.

References

1. **Siegwart, R. & Nourbakhsh, I.R. (2004).** *Introduction to Autonomous Mobile Robots*.
2. **Tzafestas, S.G., (2013).** *Introduction to Mobile Robot Control.* Elsevier, Oxford. doi: 10.1016/B978-0-12-417049-0.00025-0.
3. **El-Osery, A. & Prevost, J. (2015).** *Control and Systems Engineering: A Report on Four Decades of Contributions*. Springer, Vol. 27.
4. **Fang, Y., Liu, X., & Zhang, X. (2012).** Adaptive active visual servoing of nonholonomic mobile robots. *IEEE Transactions on Industrial Electronics,* Vol. 59, No. 1, pp. 486–497, doi: 10.1109/TIE.2011.2143380.
5. **Fu, W., Hadj-Abdelkader, H., & Colle, E. (2013).** Visual servoing based mobile robot navigation able to deal with complete target loss. *18th International Conference on Methods and Models in Automation and Robotics (MMAR),* pp. 502–507, doi: 10.1109/MMAR.2013.6669961.
6. **Liang, X., Wang, H., & Chen, W. (2014).** Adaptive image-based visual servoing of wheeled mobile robots with fixed camera configuration. *IEEE International Conference on Robotics and Automation (ICRA),* pp. 6199–6204, doi: 10.1109/ICRA.2014.6907773.
7. **Cook, G. (2011).** *Mobile Robots: Navigation, Control and Remote Sensing*. Wiley-IEEE Press, Hoboken, NJ, USA.
8. **Holland, J. (2003).** *Designing Autonomous Mobile Robots: Inside the Mind of an Intelligent Machine*. Newnes.
9. **Das, T. & Kar, I. (2006).** Design and implementation of an adaptive fuzzy logic-based controller for wheeled mobile robots. *IEEE Transactions on Control Systems Technology,* Vol. 14, No. 3, pp. 501–510, doi: 10.1109/TCST.2006.872536.
10. **Do, K., Jiang, Z., & Pan, J. (2004).** Simultaneous tracking and stabilization of mobile robots: an adaptive approach. *IEEE Transactions on Automatic Control,* Vol. 49, No. 7, pp. 1147–1151, ISSN 0018-9286, doi: 10.1109/TAC.2004.831139.
11. **Park, B.S., Yoo, S.J., Park, J.B., & Choi, Y.H. (2010).** A simple adaptive control approach for trajectory tracking of electrically driven nonholonomic mobile robots. *IEEE Transactions on Control Systems Technology,* Vol. 18, No. 5, pp. 1199–1206, doi: 10.1109/TCST.2009.2034639.
12. **Kirk, D.E. (2004).** *Optimal Control Theory: An Introduction*. Dover Publications.
13. **Basar, T. & Olsder, G.J. (1995).** *Dynamic Noncooperative Game Theory*. Academic Press, New York, NY, USA, 2 ed.
14. **Lewis, F.L. & Syrmos, V.L. (1995).** *Optimal Control*. John Wiley & Sons, Inc., New York, NY, USA, 1st ed.
15. **Al-Tamimi, A., Lewis, F., & Abu-Khalaf, M. (2008).** Discrete-time nonlinear hjb solution using approximate dynamic programming: Convergence proof. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics,* Vol. 38, No. 4, pp. 943–949, doi: 10.1109/TSMCB.2008.926614.
16. **Ohsawa, T., Bloch, A., & Leok, M. (2010).** Discrete Hamilton-Jacobi theory and discrete optimal control. *49th IEEE Conference on Decision and Control (CDC),* pp. 5438–5443, doi: 10.1109/CDC.2010.5717665.
17. **Rovithakis, G.A. & Chistodoulou, M.A. (2000).** *Adaptive Control with Recurrent High-Order Neural Networks*. Springer Verlag, Berlin, Germany.
18. **Garcia-Hernandez, R. (2005).** *Control Neuronal Descentralizado Discreto para Manipuladores Roboticos*. PhD thesis, Cinvestav, Unidad Guadalajara, Guadalajara, Jalisco, Mexico.
19. **Williams, R.J. & Zipser, D. (1989).** A learning algorithm for continually running fully recurrent neural networks. *Neural Computation,* Vol. 1, No. 2, pp. 270–280, doi: 10.1162/neco.1989.1.2.270.
20. **Leung, C.-S. & Chan, L.-W. (2003).** Dual extended Kalman filtering in recurrent neural networks. *Neural Networks*, Vol. 16, No. 2, pp. 223–239, doi: 10.1016/S0893-6080(02)00230-7.
21. **Alanis, A.Y., Sanchez, E.N., & Loukianov, A.G. (2009).** Real-time output tracking for induction motors by recurrent high-order neural network control. *17th Mediterranean Conference on Control and Automation (MED'09),* pp. 868–873, doi: 10.1109/MED.2009.5164654.

22. **Feldkamp, L.A., Prokhorov, D.V., & Feldkamp, T.M. (2003).** Simple and conditioned adaptive behavior from Kalman filter trained recurrent networks. *Neural Netw.*, Vol. 16, No. 5-6, pp. 683–689, doi: 10.1016/S0893-6080(03)00127-8.
23. **Grover, R. & Hwang, P.Y.C. (1992).** *Introduction to Random Signals and Applied Kalman Filtering*. John Wiley and Sons, NY.
24. **Song, Y., Sun, Z., Liao, X., & Zhang, R. (2006).** Memory-based control of nonlinear dynamic systems part ii- applications. *1ST IEEE Conference on Industrial Electronics and Applications,* pp. 1–6, doi: 10.1109/ICIEA.2006. 257071.
25. **Poznyak, A.S., Sanchez, E.N., & Yu, W. (2001).** *Differential Neural Networks for Robust Nonlinear Control*. World Scientific, Singapore.
26. **Felix, R.A., Sanchez, E.N., & Loukianov, A.G. (2005).** *Avoiding controller singularities in adaptive recurrent neural control*. Proceedings of the 16th IFAC World Congress, Prague, Czech Republic.
27. **Lin, W. & Byrnes, C.I. (1994).** Design of discrete-time nonlinear control systems via smooth feedback. *IEEE Transactions on Automatic Control,* Vol. 39, No. 11, pp. 2340–2346, doi: 10.1109/9.333790.
28. **Haykin, S. (2001).** *Kalman Filtering and Neural Networks*. John Wiley and Sons, NY.
29. **Hutchinson, S., Hager, G., & Corke, P. (1996).** A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation*, Vol. 12, No. 5, pp. 651–670, doi: 10.1109/70.538972**.**
30. **Espiau, B., Chaumette, F., & Rives, P. (1992).** A new approach to visual servoing in robotics. *IEEE Transactions on Robotics and Automation,* Vol. 8, No. 3, pp. 313–326, doi: 10.1109/70.143350.
31. **Chaumette, F. & Hutchinson, S. (2006).** Visual servo control, part i: Basic approaches. *IEEE Robotics and Automation Magazine*, Vol. 13, No. 4, pp. 82–90, doi: 10.1109/MRA.2006.250573.
32. **Gonzalez, R.C. & Woods, R.E. (2006).** *Digital Image Processing.* 3rd edition, Prentice-Hall Inc., Upper Saddle River, NJ, USA.
33. **Gonzalez, R.C., Woods, R.E., & Eddins, S.L. (2003).** *Digital Image Processing Using MATLAB*. Prentice-Hall Inc., Upper Saddle River, NJ, USA.
34. **Lu, F. & Milios, E. (1994).** Robot pose estimation in unknown environments by matching 2d range scans. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94),* pp. 935–938, doi: 10.1109/CVPR.1994.323928.
35. **Canudas de Wit, E., Siciliano, B., & Bastian, G. (1997).** *Theory of Robot Control*. Springer-Verlag.
36. **Salome, A., Alanis, A.Y., & Sanchez, E.N. (2011).** Discrete-time sliding mode controllers for nonholonomic mobile robots trajectory tracking problem. *8th International Conference on Electrical Engineering Computing Science and Automatic Control (CCE),* pp. 1–6, doi: 10.1109/ICEEE.2011.6106564.

Michel Lopez-Franco was born in 1986 in Guadalajara, Jalisco, Mexico. He received the B.E. in Computer Engineering from the University of Guadalajara, Guadalajara, Mexico, in 2008, and the M.Sc. in Computer and Electrical Engineering from the University of Guadalajara, Guadalajara, Mexico, in 2011. He is a Ph.D. student in Automatic Control at CINVESTAV-IPN (Advanced Studies and Research Center of the National Polytechnic Institute), Guadalajara, Mexico. His current research interests include optimal control, decentralized control, and adaptive and neural network controllers for dynamic systems.

Edgar N. Sanchez obtained the BSEE from Universidad Industrial de Santander (UIS), Bucaramanga, Colombia, in 1971, the MSEE from CINVESTAV-IPN (Advanced Studies and Research Center of the National Polytechnic Institute), Mexico City, Mexico, in 1974, and the Docteur Ingenieur degree in Automatic Control from Institut Nationale Polytechnique de Grenoble, France, in 1980. He was granted a USA National Research Council Award as a research associate at NASA Langley Research Center, Hampton, Virginia, USA (January 1985 to March 1987). His research interest centers in neural networks and fuzzy logic as applied to automatic control systems. He has been advisor of 12 Ph.D. theses and 33 M.Sc. theses. Since January 1997, he has been professor of CINVESTAV-IPN, Guadalajara Campus, Mexico.

Alma Y. Alanis, received the B.Sc. from Instituto Tecnologico de Durango (ITD), Durango Campus, Durango, Durango, in 2002, the M.Sc. and the Ph.D. in Electrical Engineering from the Advanced Studies and Research Center of the National Polytechnic Institute (CINVESTAV-IPN), Guadalajara Campus, Mexico, in 2004 and 2007, respectively. Since 2008 she has been with the University of Guadalajara, where she is currently a

ISSN 2007-9737

Chair Professor at the Department of Computer Science and a member of the Intelligent Systems Research Group. She is also a member of the Mexican National Research System (SNI-1). Her research interest centers on neural control, backstepping control, block control, and their applications to electrical machines, power systems, and robotics.

Carlos López-Franco received the Ph.D. in Computer Science in 2007 from the Center of Research and Advanced Studies, CINVESTAV, Mexico. He is currently a professor at the University of Guadalajara, Mexico, Computer Science Department, and a member of the Intelligent Systems Group. His research interests include geometric algebra, computer vision, robotics, and intelligent systems.

Article received on 24/11/2014; accepted 14/04/2015.
Corresponding author is Michel Lopez-Franco.