Feedforward Control of a Mobile Robot Using a Neural Network

Vinícius M. de Oliveira, Edson R. de Pieri
Federal University of Santa Catarina
Florianópolis, SC 88040-900 BRAZIL

Walter F. Lages
Federal University of Rio Grande do Sul
Porto Alegre, RS 90035-190 BRAZIL

Abstract

Many papers about mobile robot control deal only with the kinematic model of the robot. Since it is the kinematics and not the dynamics that introduces the nonholonomicity into the system, this approach provides a simple model while preserving the most interesting portion of the system. However, for large robots, or robots moving at high speeds, the dynamics of the body can not be neglected. This paper proposes a controller for a nonholonomic mobile robot developed by combining a kinematic controller based on nonsmooth discontinuous transformation and a dynamic controller based on a neural network (computed torque like). The overall system stability is proved by Lyapunov theory.

1 Introduction

Nowadays a large number of articles about kinematic control of nonholonomic systems can be found in the literature. Perhaps this happens due to the fact it is the kinematics that introduces the nonholonomicity into the system; moreover, it is a simpler model to consider. This approach is interesting because kinematics related parameters (geometric features) are easier to obtain than the dynamics related ones (e.g. mass, inertia moments) [1, 6, 7, 11]. However, for large robots and robots moving at high speeds, important effects are neglected. Few articles deal with the complete model of the robot, i.e., including the kinematics and the dynamics [3, 9, 2, 5].

This paper proposes a controller for a nonholonomic mobile robot developed by combining a kinematic controller and a dynamic controller. This kinematic controller is based on a discontinuous transformation [1, 8] followed by a smooth nonlinear feedback and is responsible for dealing with the nonholonomicity of the system. The other part is the dynamic controller that follows the computed torque approach based on a neural network in order to deal with the uncertain associated with the dynamic parameters.

2 Robot Model

A mobile robot system (figure 1) having an n-dimensional configuration space C with generalized coordinates \( \vec{q} = [q_1 \ldots q_n]^T \) and with m constraints can be described as follows [5], [9]:

\[
M(q)\ddot{q} + V_m \dot{q} + F(q) + \tau_d = B(q)\tau - A^T \lambda
\]

(1)

where \( M(q) \in \mathbb{R}^{n \times n} \) is the inertia matrix (symmetric and positive definite), \( V_m(q, \dot{q}) \in \mathbb{R}^{n \times n} \) is the centripetal and coriolis terms matrix, \( F(q) \in \mathbb{R}^{n \times 1} \) is the friction terms, \( \tau_d \) denotes bounded unknown disturbances including unstructured unmodeled dynamics. The matrix \( B(q) \in \mathbb{R}^{n \times r} \) is the input transformation matrix, \( \tau \in \mathbb{R}^{n \times 1} \) is the input vector, \( A(q) \in \mathbb{R}^{m \times n} \) is a matrix related with the constraints and \( \lambda \in \mathbb{R}^{m \times 1} \) is the vector of restriction forces.

Considering the time-independence of all kinematic equality constraints one can write:

\[
A(q)\dot{q} = 0
\]

(2)
Let $S(q)$ be a full rank matrix $(n - m)$ that belongs to the null space of $A^T(q)$, such that:

$$S^T(q)A^T(q) = 0$$  \hspace{1cm} (3)

Based on (2) and (3) its possible to find out a vector time function $v(t) \in \mathcal{R}^{n-m}$, for all $t$:

$$\dot{q} = S(q)v(t)$$  \hspace{1cm} (4)

The system (1) will be now transformed into a suitable representation to the control perspective. Differentiating (4) and replacing the result in (1), then pre-multiplying by $S^T(q)$ and using (2) and (3) it is possible to eliminate the constraint matrix $A^T(q)\lambda$, resulting in:

$$S^TMS\dot{\nu} + S^T(MS + +V_mS)v + \dot{S}^T F + S^T r_d = S^TB\tau$$  \hspace{1cm} (5)

The expression can be rewritten as follows:

$$\bar{M}\dot{v} + \bar{V}_m v + \bar{F} + \bar{r}_d = \bar{\tau}$$  \hspace{1cm} (6)

where $\bar{M}(q) \in \mathcal{R}^{r \times r}$ is the inertia matrix (symmetric and positive definite), $\bar{V}_m(q, \dot{q}) \in \mathcal{R}^{r \times r}$ is the centripetal and coriolis terms matrix, $\bar{F}(v) \in \mathcal{R}^{r \times 1}$ is the friction terms, $\bar{r}_d$ denotes bounded unknown disturbances including unstructured unmodeled dynamics. $\bar{\tau}$ is the input vector ($\bar{B}$ is a constant nonsingular matrix that depends on geometric parameter of the robot). The matrix $\bar{M}$ and the norm of the $\bar{V}_m$ are bounded and the matrix $\bar{M} - 2\bar{V}_m$ is skew-symmetric, used to prove the stability of the system by Lyapunov method.

### 3 Control Structure

The proposed control structure is shown in figure 2. It is possible to observe that no previous knowledge of the dynamics of the robot, (6), is necessary since it is the role of the neural network to learn the dynamics on-the-fly.

In the literature, the problem is simplified by neglecting part of the dynamics of the system, considering only the part described by (4).

#### 3.1 Kinematic Controller

In this section a coordinate transformation is presented. Consider the mobile robot in figure 1 and from (4) the kinematics is given by:

$$\begin{bmatrix}
\dot{x}_e \\
\dot{y}_e \\
\dot{\theta}_e
\end{bmatrix} =
\begin{bmatrix}
\cos(\theta) & 0 & v \\
\sin(\theta) & 0 & \omega \\
0 & 1 & 0
\end{bmatrix}
\begin{bmatrix}
v \\
\omega
\end{bmatrix}$$  \hspace{1cm} (7)

where $v$ is the linear velocity and $\omega$ is the angular velocity of the robot. Now, applying the nonsmooth transformation to the system (7) results:

$$e = \sqrt{x_e^2 + y_e^2}$$  \hspace{1cm} (8)

$$\phi = \text{atan2}(y_e, x_e)$$  \hspace{1cm} (9)

$$X_e = e\cos(\phi)$$  \hspace{1cm} (10)

$$Y_e = e\sin(\phi)$$  \hspace{1cm} (11)

Differentiating expressions (8) and (9) and using ex-
pressions (10) and (11) it results:
\[
\begin{bmatrix}
\dot{e} \\
\dot{\phi} \\
\dot{\alpha}
\end{bmatrix} =
\begin{bmatrix}
\cos(\phi) & \sin(\phi) \\
-\frac{e}{\alpha} & \cos(\phi)
\end{bmatrix}
\begin{bmatrix}
v \\
\omega
\end{bmatrix}
\] (12)

Let \(\alpha \triangleq \theta - \phi\). Taking its derivative and using expressions (7) and (12) one can express the kinematics of the mobile robot in polar coordinates as:
\[
\begin{bmatrix}
\dot{e} \\
\dot{\phi} \\
\dot{\alpha}
\end{bmatrix} =
\begin{bmatrix}
\cos(\phi) & 0 & 0 \\
\frac{e}{\alpha} & \cos(\phi)
\end{bmatrix}
\begin{bmatrix}
v \\
\omega
\end{bmatrix}
\] (13)

Consider \(V_1\) as a Lyapunov function candidate, expressed by:
\[
V_1(e, \phi, \alpha) = \frac{1}{2}(e^2 + \alpha^2 + h\phi^2)
\] (14)

where \(h\) is a positive constant.

Taking the first derivative of (14), one can define the control velocities as follows:
\[
\begin{bmatrix}
v \\
w
\end{bmatrix} =
\begin{bmatrix}
-\gamma_1 e \cos(\phi) \\
-\gamma_2 \alpha - \gamma_1 \cos(\phi) \sin(\alpha) / \alpha (\alpha - h \phi)
\end{bmatrix}
\] (15)

and it leads to:
\[
\dot{V}_1 = -\gamma_1 e^2 \cos(\alpha)^2 - \gamma_2 \alpha^2 \leq 0
\] (16)

From (16) and from the fact that \(V_1\) is continuous and nonnegative, the closed loop system is stable. Thus, as \(V_1\) is uniformly continuous and bounded and by Barbalat lemma, one has that \(e\) and \(\alpha\) converge to zero. Then, the convergence of \(\phi\) to zero can be proved by considering the closed-loop system equations:
\[
\begin{align*}
\dot{e} &= -\gamma_1 e \cos(\alpha)^2 \\
\dot{\phi} &= -\gamma_1 \sin(\alpha) \cos(\alpha) \\
\dot{\alpha} &= -\gamma_2 \alpha + \gamma_1 h \phi \cos(\alpha) \sin(\alpha) / \alpha
\end{align*}
\] (17)

As \(\alpha\) converges to zero, from the closed-loop system (17) it is possible to note that \(\dot{\alpha}\) converges to some constant value given by \(\gamma_1 h \phi^*\). However, the uniform continuity of \(\dot{\alpha}\), along with the convergence to zero of \(\alpha\), ensures, by the Barbalat lemma, that \(\dot{\alpha}\) converges to zero, forcing \(\phi^*\) to be zero.

3.2 Neural Network

In this work a multi-layered perceptron feedforward neural network, using an on-line weight tuning is considered.

This neural network has an input layer with 6 neurons, a hidden layer with 8 neurons and, the output layer has 2 neurons. The activation function is the well-known sigmoid activation function:
\[
\sigma(a) = \frac{1}{1 + e^{-a}}
\] (18)

The output of the neural net is a vector \(y \in \mathbb{R}^{2 \times 1}\), expressed by:
\[
y(x) = W^T \sigma(V^T x)
\] (19)

where \(V \in \mathbb{R}^{6 \times 8}\) is the weight matrix between the input and the hidden layer, \(W \in \mathbb{R}^{8 \times 2}\) is the weight matrix between the hidden layer and the output layer and \(x \in \mathbb{R}^{6 \times 1}\) is the input vector of the neural network.

One of the most important features of a neural network is its capability to approximate multivariate nonlinear continuous function [10]. Considering this fact, let \(f(x)\) be a smooth function from \(\mathbb{R}^n \rightarrow \mathbb{R}^m\), so it is possible to show that, as long as \(x\) is restricted to a compact set \(U_n \subset \mathbb{R}^n\), for some number \(N\) of hidden layer neurons, there is a neural network configuration (weights and thresholds) such that:
\[
f(x) = \hat{W}^T \sigma(\hat{V}^T x) + \epsilon
\] (20)

The expression (20) denotes that a neural network can approximate any continuous function in a compact set. The error in the neural net approximation is denoted by \(\epsilon\). For any specified value to \(\epsilon_N > 0\), is possible to find \(\epsilon < \epsilon_N\). In the control perspective the meaningful concept is that, specifying \(\epsilon_N\), there exist some combination of weights such that the maximum desired approximation error can be obtained. Thus, an estimate of \(f(x)\) is given by:
\[
\hat{f}(x) = \hat{W}^T \sigma(\hat{V}^T x) + \epsilon
\] (21)

where \(\hat{V}\) and \(\hat{W}\) are estimates of the ideal weights of the neural network.

The weights are adjusted on-line by the following expressions:
\[
\Delta \hat{W} = F \sigma'(\hat{V}^T x) e_c^T - k \hat{F} e_c \parallel \hat{W}
\] (22)
\[
\Delta \hat{V} = G \sigma'(\hat{V}^T x) e_c \parallel \hat{V}
\] (23)

where the design parameters \(F\) and \(G\) are positive definite matrices and \(k > 0\).
3.3 Feedforward Control

Using the velocity control from the kinematic controller it is possible to express the velocity error as:

\[ e_c = v_c - v \]  \hspace{1cm} (24)

Thus, taking the time derivative of (24) one can rewrite the dynamics by using the velocity tracking error as:

\[ \dot{M} \dot{e}_c = - \nabla_m e_c - \ddot{v} + f(x) + \ddot{v}_d \]  \hspace{1cm} (25)

with

\[ f(x) = \dot{M} \dot{v}_c + \nabla_m v_c + \bar{F}(v) \]  \hspace{1cm} (26)

Due to the absence of a complete knowledge of the parameters involved, the neural network is introduced to perform the nonlinear mapping as intended in (26). The input vector of the neural net is given by:

\[ x = [v_c \quad \dot{v}_c \quad v]^T \]  \hspace{1cm} (27)

A suitable control input for velocity following is a computed-torque like control:

\[ \ddot{v} = \hat{f} + K_4 e_{\dot{c}} - \gamma \]  \hspace{1cm} (28)

where \( K_4 \) is a positive definite gain matrix and \( \hat{f}(x) \) is an estimative of (26) given by the neural network. The signal \( \gamma \) is used to ensure robustness. Replacing (15) in (25) the closed-loop dynamic error is determined by:

\[ \dot{M} \dot{e}_c = -(K_4 + \nabla_m)e_c + \hat{f}(x) + \ddot{v}_d + \gamma \]  \hspace{1cm} (29)

where \( \hat{f} = f - \dot{f} \). The overall system stability is guaranteed by Lyapunov theory and robustness aspects are presented in [5].

Consider the following Lyapunov function candidate:

\[ V = V_1 + V_2 \]  \hspace{1cm} (30)

where \( V_1 \) is defined in expression (14) and

\[ V_2 = \frac{1}{2} \dot{e}_c^T \dot{M} \dot{e}_c + tr\{\dot{W}^T F^{-1} \dot{W}\} + tr\{\dot{V}^T G^{-1} \dot{V}\} \]  \hspace{1cm} (31)

Taking the time-derivative of (30) results:

\[ \dot{V} = e_c \dot{e}_c + h \ddot{\phi} + \alpha \ddot{\alpha} + \dot{V}_2 \]  \hspace{1cm} (32)

and differentiating (31), then by replacing (29) in (31) and using the skew-symmetry property gives:

\[ \dot{V}_2 = -e_c^T K_4 e_c + e_c^T (\delta + \gamma) + k e_c \| \dot{Z}(Z - \hat{Z}) \| \]  \hspace{1cm} (33)

where \( \delta \) is the disturbance term expressed \(^1\) as:

\[ \delta = \dot{W} \dot{\sigma}^T \dot{V} x + \dot{W}^T O(\dot{V}^T x) + \epsilon + \ddot{v}_d \]

Now considering that

\[ tr\{\dot{Z}(Z - \hat{Z})\} = < \dot{Z}, Z > - \| \dot{Z} \| ^2 \leq || \dot{Z} || \| Z \| - \| \dot{Z} \| \]  \hspace{1cm} (34)

then expression (33) can be rewritten as:

\[ \dot{V}_2 \leq -e_c^T e_c - \| e_c \| \left( k \| \dot{Z} \| \left( \| \dot{Z} \| - Z_M \right) + K_4 \| e_c \| - C_0 - C_1 \| \dot{Z} \| \right) \]  \hspace{1cm} (35)

Using expression (16) to the first three terms of (32) and replacing \( \dot{V}_2 \) from (32) yields:

\[ \dot{V} \leq \gamma - \gamma_1 e_c^2 \cos(\alpha)^2 - \gamma_2 \alpha^2 + k \| e_c \| \left( K_4 \| e_c \| - C_0 + C_1 \| \dot{Z} \| + k \| \dot{Z} \| \left( \| \dot{Z} \| - Z_M \right) \right) \]  \hspace{1cm} (36)

Thus, to \( \dot{V} \) be nonpositive, the terms in braces must be nonnegative. Let \( C_3 \) be defined as \( C_3 \triangleq (1/2)(Z_M + (C_1/k)) \) and completing square in (36) results:

\[ \dot{V} \leq V_1 - \| e_c \| \left( K_4 \| e_c \| + k \left( \| \dot{Z} \| - C_0 \right) ^2 - C_0 - k C_3 ^2 \right) \]  \hspace{1cm} (37)

Thus, \( \dot{V} \) is negative as long as

\[ \| e_c \| > \frac{k C_3 ^2 + C_0}{k} \]  \hspace{1cm} (38)

or

\[ \| \dot{Z} \| > C_0 + \sqrt{C_3 ^2 + C_0} \]  \hspace{1cm} (39)

which guarantees that \( \| e_c \| \) and \( \| \dot{Z} \| \) are locally uniformly bounded.

\(^1O(\dot{V}^T x)\) denotes the higher order terms in the Taylor series.
4 Results

In this section the results obtained by two different real-time simulations are presented. The first simulation is a parking maneuver and the second one is a path following task. The kinematic controller gains are $\gamma_1 = 0.09$, $\gamma_2 = 0.7$ and $h = 7.0$ for the first simulation and $\gamma_1 = 0.9$, $\gamma_2 = 0.05$ and $h = 7.0$ for the second one. The neural network sigmoidal activation function is used for the 8 hidden-layer neurons, $F = G = 4 \times 10^{-3}I$, $k = 10^{-2}$, $K_4 = 10 \times I$.

4.1 Parking

The reference posture is $X = 2.0m$, $Y = 1.0m$ and $\theta = \pi$ rad. The spatial trajectory followed by the robot to complete the parking is shown in figure 3. The initial position is $X_0 = -3.0$, $Y_0 = 2.0$ and $\theta = \pi$. The torques developed by the motors are presented in figure 4(a). Figures 4(b) and 4(c) show the errors in the $X$ and $Y$ directions and the error in the orientation, respectively.

![Figure 3: Robot trajectory for a parking maneuver.](image)

Figure 4: (a) Torque on the right motor (solid) and on the left motor (dashed) for parking maneuver; (b) errors in $X$ (solid) and $Y$ (dashed) directions; (c) error in the orientation of the robot.

4.2 Path Tracking

This second simulation is a path tracking task with reference trajectory given by $X = 0.2t + 3.0$ and $Y = 0.2t - 2$. Figure 5 shows the complete movement of the path tracking maneuver. The initial position is $X_0 = 0.5$, $Y_0 = -1.0$ and $\theta = 0.0$. Figure 6(a) shows the torques developed by the motors. The trajectory tracking errors are presented in figures 6(b) and 6(c).

5 Conclusion

In this paper a neural network is used in order to cope with the incomplete knowledge of the dynamics and physical parameters (e.g. friction coefficient) of a mobile robot. It is important to note that the neural network can not have a large number of neurons in the hidden-layer in order to respect the time-restrictions. Since the simulations were performed under real-time, these restrictions were checked.

The controller proposed in this work does not need a reference velocity signal, as opposed to the one proposed by [5].

The next step is validate the proposed controller under real problems such as error from sensors, slipping, etc.
Figure 5: Reference trajectory (solid) and actual trajectory (dashed) for a path tracking maneuver.

References


