Neural network-based $\mathcal{H}_\infty$ control for fully actuated and underactuated cooperative manipulators

Adriano A.G. Siqueira a,*, Marco H. Terra b

a Department of Mechanical Engineering, University of São Paulo at São Carlos, Av. Trabalhador Sãocarlense, 400 - 13566-590, São Carlos-SP, Brazil
b Department of Electrical Engineering, University of São Paulo at São Carlos, Brazil

ARTICLE INFO

Article history:
Received 29 March 2007
Accepted 12 September 2008

Keywords:
Cooperative manipulators
Robust control
Neural control

1. Introduction

The problem of robust control for cooperative manipulators rigidly connected to solid objects is addressed in this paper. Cooperative manipulators have been built to be applied in several areas of research and in diverse industrial applications. The advantages of this kind of robotic architecture can be summarized as improving the load capacity of the robotic system (Torres, Santiago, & Díaz, 2008). Design paradigms to solve force/position control problems have been established in the literature to improve the performance of the cooperation. In Wen and Kreutz-Delgado (1992) a control strategy for cooperative manipulators was proposed based on the independence of the position and force controls. The applied force between the manipulator end-effectors and the object is decomposed into motion force and squeeze force, which must be controlled. In Tinós and Terra (2006), the hybrid position/force controller proposed in Wen and Kreutz-Delgado (1992) is extended to underactuated cooperative manipulators.

An underactuated cooperative architecture can be characterized in two possible ways: when it is caused by failures in the actuators or when the manipulators are specifically designed to take into account the underactuation as a structural concept (Tinós, Terra, & Bergerman, 2007). In the latter case, one can obtain lighter or less bulky structures. However, it is clear that for this kind of robot the robustness against uncertainties and external disturbances naturally decreases if an appropriate robust control strategy is not applied. Moreover, due to the fact that some degrees of actuation are lost in the underactuated case, the squeeze force control can be performed only in some components of the squeeze force.

However, according to Tinós and Terra (2006), in the case of underactuated cooperative manipulators it is important to recalculate the dynamic load-carrying capacity (DLCC) of the system. When the robots lose one or more actuators, the DLCC generally decreases. The DLCC is defined as the maximum load that can be carried by the system in a specified trajectory. The details of how to compute the DLCC of this class of system can be found in Tinós and Terra (2006).

In Siqueira and Terra (2007b), two nonlinear $\mathcal{H}_\infty$ control techniques based on centralized control strategies are evaluated for underactuated cooperative manipulators: $\mathcal{H}_\infty$ control for linear parameter varying (LPV) systems (Wu, Yang, Packard, & Becker, 1996) and $\mathcal{H}_\infty$ control based on game theory (Chen, Lee, & Feng, 1994). These controllers are applied considering the control strategy proposed in Wen and Kreutz-Delgado (1992), where the squeeze force control is designed independently of the position control. In this case, the $\mathcal{H}_\infty$ performance index deals only with the position control problem. A semi-decentralized adaptive fuzzy-based controller with $\mathcal{H}_\infty$ performance is developed for fully actuated cooperative manipulators in Lian, Chiu, and Liu (2002). For this approach, the $\mathcal{H}_\infty$ performance index includes the position and squeeze force errors.

In this paper, a neural network-based $\mathcal{H}_\infty$ control is developed for fully actuated and underactuated manipulators (Siqueira & Terra, 2007a). The neural network approximates only the
uncertainties of the model, since the nominal model is assumed to be well known. As in Lian et al. (2002), the $\mathcal{H}_\infty$ performance index includes the position and squeeze force errors, which guarantees an overall disturbance rejection. Furthermore, for underactuated manipulators, a practical solution is proposed for the squeeze force control. In this case, only some components of the squeeze force can be controlled and constraints are imposed on the components that are not controlled. The proposed approach guarantees asymptotic convergence of the motion tracking errors in spite of parametric uncertainties and external disturbances. An underactuated cooperative manipulator is used to obtain experimental results in order to confirm the efficiency of the proposed approach.

This paper is organized as follows: Section 2 shows the dynamic equations for fully actuated manipulators; Section 3 develops the neural network-based $\mathcal{H}_\infty$ control approach for cooperative manipulators; Section 4 presents the dynamic equations for underactuated manipulators and the squeeze force control problem; and Section 5 provides the experimental results.

2. Fully actuated cooperative manipulators

Consider a cooperative manipulator consisting of $m$ fully actuated manipulators, where each one has $n$ degrees of freedom. Let $q_i \in \mathbb{R}^n$ be the vector of generalized coordinates of manipulator $i$ and $x_o \in \mathbb{R}^n$ be the vector composed of Cartesian coordinates and orientation of the object, which are rigidly connected to the end-effectors of the manipulators. The geometric constraints, generated by this configuration, are given by $q_i(x_o, q_i) = 0$ for $i = 1, 2, \ldots, m$. Denote $J_{eq}(x_o, q_i)$ and $J(x_o, q_i)$ the Jacobian matrices of $q_i(x_o, q_i)$ with relation to $x_o$ and $q_i$, that is, $J_{eq}(x_o, q_i) = \partial q_i/\partial x_o$ and $J(x_o, q_i) = \partial q_i/\partial q_i$, respectively. Hence, the velocity constraints are given by $\dot{q}_i(x_o, q_i) = J_{eq}(x_o, q_i) \dot{x}_o + J(x_o, q_i) \dot{q}_i; \dot{q}_i = 0$, for $i = 1, 2, \ldots, m$. Assume that the relation $\ddot{q}_i = J_{eq}^T(x_o, q_i) J(x_o, q_i) \dot{q}_i$, for $i = 1, 2, \ldots, m$, can always be computed. Details on the invertibility of the Jacobian matrix $J$ for all $x_o$ and $q_i$ can be seen in Craig (2004) and in Caccavale, Chiaccio, and Chiaverini (1999). With this result, the kinematic constraints can be written as

$$\dot{q}_i = \left[ \begin{array}{c} \dot{x}_o \\ -J_{eq}^{-1}(x_o) J(x_o) \end{array} \right] \dot{q}_i = B(x_o) \dot{q}_i,$$

(1)

where $B(x_o) = \left[ \begin{array}{cc} \dot{x}_o \\ -J_{eq}^{-1}(x_o) J(x_o) \end{array} \right] \in \mathbb{R}^{2n \times n}$. The resulting force on the object is given by $F(x_o, \dot{q}_i) = M(x_o) \ddot{q}_i + C(x_o, \dot{q}_i, \ddot{q}_i) \dot{q}_i + g(x_o) + \tau_d = \tilde{F}(x_o, \dot{q}_i),$

(2)

where $M(x_o)$ is the inertia matrix, $C(x_o, \dot{q}_i, \ddot{q}_i)$ is the Coriolis and centripetal matrix, $g(x_o)$ is the gravitational torque vector, and $\tau_d$ is the disturbance force vector on the object, and $h = [h^T \cdots h^T]$ is the vector of applied force by the manipulator $i$ in the object.

The dynamic equation of the manipulator $i$ is given by

$$M(q_i) \ddot{q}_i + C_i(q_i, \dot{q}_i, \ddot{q}_i) \dot{q}_i + g_i(q_i) + \tau_d = \dot{J}^T(x_o, \Omega(x_o)) h,$$

(3)

where $M(q_i)$ is the inertia matrix, $C_i(q_i, \dot{q}_i, \ddot{q}_i)$ is the Coriolis and centripetal matrix, $g_i(q_i)$ is the gravitational torque vector, $\tau_d$ is the disturbance torque vector, and $\tau$ is the applied torque vector, of manipulator $i$. Then, the dynamic equation of the cooperative manipulator can be represented as

$$M(\theta) \ddot{q} + C(\theta, \dot{\theta}) \dot{q} + g(\theta) + \tau_d = \frac{\tau}{T} + \tilde{f}(x_0, \dot{x}_0) h,$$

(4)

where $M(\theta) = \text{diag}(M_0(x_o), M_1(q_1), \ldots, M_m(q_m))$, $C(\theta, \dot{\theta}) = \text{diag}(C_0(x_0, q_0), C_1(q_1), \ldots, C_m(x_0, q_m))$, $g(\theta) = \left[ g_{\text{cm}}(x_0)^T \; g_{\text{cm}}(q_1)^T \; \cdots \; g_{\text{cm}}(q_m)^T \right]^T$, $\tau_d = \left[ \tau_1^T \; \cdots \; \tau_m^T \right]^T$, and $\tau = \left[ \tau^T \; \cdots \; \tau^T \right]^T$.

Let $h_o$ be the projection of $h$ on the object frame $C$ (frame fixed on the center of mass of the object), $h_o = J_{eq}(x_o) h$, with $J_{eq}(x_o) = \text{diag}(J_{eq}(x_o, q_1), \ldots, J_{eq}(x_o, q_m))$. The resulting force on the object, $h_o = \tilde{f}(x_0) h_o$, can be rewritten as $h_o = A^T \tilde{f}(x_0) h_o = A^T h_o$, where $A^T = \left[ I_n \; I_n \; \cdots \; I_n \right] \in \mathbb{R}^{n \times (m-1)}$ and $I_n$ is the identity matrix of size $n$. Hence, $A^T$ transforms the $nm$-dimensional vector $h_o$ into the $n$-dimensional vector of the resulting force at the object frame $C$. According to Wen and Kreutz-Delgado (1992), the matrix $A^T$ is used instead of $\tilde{f}(x_0)$ in the following definitions to avoid uniqueness problems.

Since $A^T$ is a non-square and full row-rank matrix, there exists a non-trivial null space, denoted the squeeze subspace $X_s$, given by $X_s = \{h_s \in \mathbb{R}^m | A^T h_s = 0 \}$. The dimension of $X_s$ is $n(m-1)$. If $h_o$ belongs to the null space $X_s$, the resulting force has no contribution to the object movement. Define the following orthogonal decomposition of the projection of the applied force: $h_o = h_{ps} + h_{psm}$, where $h_{ps}$ is the projection of $h_o$ in $X_s$, the squeeze force, and $h_{psm}$ are the forces induced by the system movement, the motion force.

Considering the orthogonal decomposition, the dynamic equation of the cooperative manipulator, (4), can be represented as

$$M(\theta) \ddot{q} + C(\theta, \dot{\theta}) \dot{q} + g(\theta) + \tau_d = \tau_v + A^T h_{psm},$$

(5)

where $\tau_v$ is an auxiliary control input,

$$\tau_v = \left[ \begin{array}{c} A^T \tilde{f}(x_0) \dot{h}_0_m \end{array} \right]$$

and $A^T h_{psm} = \left[ A^T \tilde{f}(x_0) \dot{h}_0_m \right] \in \mathbb{R}^{n \times (m-1)}$ is a Jacobian matrix. If the auxiliary control input is partitioned in two vectors, $\tau_{v1} = A^T \tilde{f}(x_0)$ and $\tau_{v2} = \tau + \tilde{f}(x_0) \dot{h}_0_m$, the applied torque vector can be computed by $\tau_v = \tau_{v2} - \tilde{f}(x_0) \dot{h}_0_m (A^T)^{-1} \tau_{v1}$, where $(A^T)^{-1}$ is the pseudo-inverse of $A^T$. The motion force is given by $h_{psm} = (A^T)^{-1} \tau_{v1}$. Hence, the control problem is to find an auxiliary control in order to guarantee stability and robustness against disturbances.

With the kinematic constraints (1), one obtains

$$M(x_o) \dot{x}_o + C(x_o, \dot{x}_o) \ddot{x}_o + \left( M(x_o) \ddot{q}_i + C_i(q_i, \dot{q}_i, \ddot{q}_i) \dot{q}_i + g_i(q_i) + \tau_d \right) = \tau_v + A^T \tilde{f}(x_0) h_{psm},$$

(6)

Consider a bounded desired trajectory for the object, $x_{d}^T \in \mathbb{R}^m$, and a bounded desired squeeze force, $h_{psm}^T \in \mathbb{R}^m$. Define the following auxiliary variable:

$$\eta = B(x_o) (A^T \tilde{f}(x_0) \dot{h}_0_m - \eta \tilde{E}_2 e_f),$$

where $\eta \in \mathbb{R}^{n(m-1)}$, $e_f = x_d^T - x_o$. $A_0$ is a symmetric positive definite matrix, $\eta > 0$, $E_2 = \left[ 0 \right]_{nm \times nm}$, and $\tilde{E}_2 \in \mathbb{R}^{nm \times mm}$ is the output of the stable filter

$$\tilde{e}_f + \eta \tilde{e}_f = -\tilde{A}_2 \tilde{E}_2^T \tilde{A}_2^T \eta \tilde{E}_2 h_{psm},$$

(7)

with a symmetric positive definite matrix $\tilde{A}_2$ and $\tilde{E}_2 = \tilde{E}_2^T - \tilde{E}_2$, the squeeze force error. A composed error signal can now be defined as

$$s = \tau_v - B(x_o) \dot{x}_o = B(x_o) (A^T \tilde{f}(x_0) \dot{h}_0_m + e_f - \eta \tilde{E}_2 e_f),$$

Please cite this article as: Siqueira, A. A. G., & Terra, M. H. Neural network-based $\mathcal{H}_\infty$ control for fully actuated and underactuated cooperative manipulators. Control Engineering Practice (2008), doi:10.1016/j.conengprac.2008.09.007.
with \( s \in \mathbb{R}^{m+1} \). Other representations of the error measure can be obtained by applying a stable filter (7) and defining the error terms \( e_1 = \text{col}(e_{o_1,1}, \ldots, e_{o_1,m}) \) and \( e_2 = \text{col}(e_{o_2,1}, \ldots, e_{o_2,l}) \):

\[
\begin{align*}
s &= L(e_1 + e_2) + E_1(\dot{f}(x) + \hat{f}(x) - \hat{f}(x))
L &= \text{diag}(\lambda_1, \ldots, \lambda_l), \quad E_1 = [e_{o_1} \ldots e_{o_1} e_{o_2} \ldots e_{o_2}]
\end{align*}
\]

(8)

where \( L = [B(x_0) E_2] \in \mathbb{R}^{(m+1)\times(m+1)} \), \( E_1 = [\mu_0 \ 0]^T \in \mathbb{R}^{(m+1)\times n} \), and \( \lambda = \text{diag}(\lambda_1, \ldots, \lambda_l) \). The error terms, \( e_1 \) and \( e_2 \), yield the composed error \( e = [e_1^T \ e_2^T]^T \). From (6) and the above error measure, \( s \), the error dynamics is given by

\[
M(x_e) \ddot{s} = M(x_o) \ddot{\theta} - M(x_o) \dot{B}(x_0) \dot{x}_o + B(x_0) \ddot{x}_o
\]

(9)

where \( x_o = [x_o \ x_o^T \ q_r^T \ \dot{q}_r^T]^T, \ F_0(x_o) = M(x_o) \dot{x}_o + C(x_o, \dot{x}_o) \dot{q}_r + g(x_o) \), and \( \Delta F(x_o) = M(x_o) \Delta \dot{x}_o + C(x_o, \dot{x}_o) \Delta \dot{q}_r + \Delta g(x_o) \).

The terms \( M(x_o), C(x_o, \dot{x}_o), \) and \( g(x_o) \) represent nominal values for matrices \( M(x_o), C(x_o, \dot{x}_o), \) and \( g(x_o) \), respectively. The parametric uncertainties are represented by \( \Delta M(x_o), \Delta C(x_o, \dot{x}_o), \) and \( \Delta g(x_o) \).

3. Neural network-based \( \mathcal{H}_\infty \) controller

In this section, a set of \( k, (k = 1, \ldots, n) \) neural networks \( \Delta F(x_e, \Theta_k) \), where \( \Theta \) is a vector containing the tunable network parameters, is used to approximate only the uncertain term \( \Delta F(x_e) \) in (9). Each neural network is composed of nonlinear neurons in the hidden layer and linear neurons in the input and output layers, with adjustable parameters \( \Theta_k \) in the output layers (Chang, 2000; Chang & Chen, 1997). The single-output neural networks are of the form

\[
\Delta F_k(x, \Theta_k) = \sum_{i=1}^{n_k} H_i \left( \sum_{j=1}^{m_k} w_{ij} k_2 \right) \Theta_{ki} \Theta_k = \phi_k^T \Theta_k
\]

(10)

where

\[
\begin{bmatrix} H_k \end{bmatrix} = \begin{bmatrix} \ldots \end{bmatrix}, \quad \Theta_k = \begin{bmatrix} \Theta_{ki} \end{bmatrix}
\]

and \( p_k \) is the number of neurons in the hidden layer. The weights \( w_{ij} \) and the biases \( m_k \) for \( 1 \leq i \leq n_k, 1 \leq j \leq 5m \), and \( 1 \leq k \leq n \) are assumed to be constant and specified by the designer. \( H(.) \) is selected to be a hyperbolic tangent function. The complete neural network can be denoted by

\[
\Delta F(x, \Theta) = \begin{bmatrix} \Delta F_1(x_1, \Theta_1) \\ \vdots \\ \Delta F_n(x_n, \Theta_n) \end{bmatrix} = \begin{bmatrix} \phi_1^T \Theta_1 \\ \vdots \\ \phi_n^T \Theta_n \end{bmatrix} = \phi^T \Theta
\]

(11)

The following assumptions are considered (Chang, 2000):

(i) There exists an optimal parameter value \( \Theta^* \in \Omega_\Theta \) such that \( \Delta F(x_e, \Theta^*) \) approximates \( \Delta F(x_e) \) as closely as possible, where \( \Omega_\Theta \) is a pre-assigned constraint region.

(ii) The approximation error, \( \delta F(x_e) = \Delta F(x_e) - \Delta F(x_e, \Theta^*) \), must be bounded by a state-dependent function; that is, there exists a function \( k(x_e) > 0 \) such that \( |\delta F(x_e)| < k(x_e) \), for all \( 1 \leq i \leq n \).

Considering the above assumptions, the error dynamics can be rewritten as

\[
\begin{align*}
M(x_e) \ddot{s} &= - C(x_e, \dot{x}_e) + F_0(x_e) + \Delta F(x_e, \Theta^*) + \delta F(x_e)
L &= \text{diag}(\lambda_1, \ldots, \lambda_l), \quad E_1 = [e_{o_1} \ldots e_{o_1} e_{o_2} \ldots e_{o_2}]
\end{align*}
\]

(12)

With these assumptions, the adaptive neural network nonlinear \( \mathcal{H}_\infty \) control problem for cooperative manipulators can be formulated as follows: given the level of attenuation \( \gamma \), find an auxiliary control input \( \tau_v \) such that the following \( \mathcal{H}_\infty \) performance index is achieved

\[
\begin{align*}
\int_0^T s^T \Psi_s \, dt &= \int_0^T \varepsilon^T Q \varepsilon \, dt \\
+\tau_v^T (Z \tau_v) + \tau_v^T \int_0^T \tau_v^T \, dt \label{eq:13}
\end{align*}
\]

(13)

where

\[
\Psi = \begin{bmatrix} A^T \Psi A & A^T \Psi \\ \Psi A & \Psi \end{bmatrix}
\]

with \( \Psi = \Psi^T > 0 \) and \( A = \text{diag}(A_0, A_1) \). \( e \) is the composed error, \( Z \) is a symmetric positive definite matrix, \( \Theta = \Theta^* - \Theta \) denotes the neural parameter estimation error and \( \tau_v \) is a square-integrable torque disturbance, i.e., \( \tau_v \in L_2 \) (Zhou, Doyle, & Glover, 1996 contains details on square-integrable signals).

Theorem 1. Consider a cooperative manipulator as described by (4). If the control law is defined as

\[
\dot{\Theta} = \text{Proj}[S^T Z S]\]

(15)

\[
\tau_v = F_0(x_e) + K_s + Z \Theta - R\ddot{x}_e - \rho \dot{R}(x_e) \dot{h}_o + \frac{p}{n} R^T(x_e) \dot{h}_o + \tau_s,
\]

(16)

with \( \tau_s = k(x_e) \text{sgn}(s) \) and \( \text{Proj}[S^T Z S] \)

\[
\begin{cases} S^T Z S & \text{if } \Theta \Theta^* < M \text{ or } (\Theta^* \Theta) > M \text{ and } \Theta^* S^T Z S \Theta \leq 0, \\
S^T Z S - \frac{(\Theta^* \Theta - M) \Theta^* S^T Z S \Theta}{\Theta^* \Theta} & \text{otherwise,}
\end{cases}
\]

where \( K = \text{diag}(K_o, K_1, \ldots, K_n) \), with symmetric positive definite matrices \( K_o, K_1, \ldots, K_n \), and \( \text{Proj}[S^T Z S] \) is a projection algorithm, then the closed-loop error system satisfies:

(1) \( e_o, \dot{e}_o, \) and \( \ddot{e}_o \in L_\infty, e_1, \) and \( \dot{h}_o \in L_\infty, \) and \( \Theta \in \Omega_\Theta \),

(2) The \( \mathcal{H}_\infty \) performance index (13) is achieved if \( K_i \) is selected as \( K_i = P_i + (1/A_i)I_n \) with symmetric positive definite matrix \( P_i \),

(3) If \( d \in L_2 \), then the motion tracking errors \( e_o, \dot{e}_o \) converge to zero as \( t \to \infty \).

A proof of this theorem can be found in Appendix A.

Remark 1. The algorithm \( \text{Proj}[S^T Z S] \) was originally defined in Khalil (1996). It guarantees that \( \Theta(t) \in \Omega_\Theta \) for all \( t \) (where \( \Omega_\Theta = \{ \Theta : \Theta^* \Theta < M + \delta \} \), for some \( M > 0 \) and \( \delta > 0 \) is a pre-assigned constraint region for \( \Theta \)).

4. Underactuated cooperative manipulators

Consider now that the joints of the cooperative manipulator are formed by \( n_a \) active joints (with actuators) and \( n_p \) passive joints (without actuators). The kinematic constraints (1) can be
rewritten as
\[
\dot{x}_0 = \left[ \begin{array}{c} I_n \\ -J_{AP}(x_0)A(x_0) \end{array} \right] \tilde{\Phi}(x_0) \dot{x}_0 + \tilde{\Phi}(x_0) \dot{x}_0, \tag{17}
\]
where \( \tilde{\Phi} = [\Phi_1^T, \Phi_2^T, \ldots, \Phi_m^T]^T \), \( \Phi_q \in \mathbb{R}^{m \times 1} \) is the position vector of active joints, \( \Phi_0 \in \mathbb{R}^{m \times 1} \) is the position vector of passive joints and \( J_{AP}(x_0) \) is a Jacobian matrix generated from the orthogonal permutation of the load position \( \Phi_0 \).

The dynamic equation of underactuated cooperative manipulators can be given by
\[
\dot{\mathbf{M}}(\dot{\theta}) \ddot{\theta} + \mathbf{C}(\dot{\theta}, \dot{\theta}) \dot{\theta} + \mathbf{g}(\dot{\theta}) + \tau_d = \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} J_{AP}^T(x_0) \\ J_{AP}^T(x_0) \end{bmatrix} \tau_n, \tag{18}
\]
where \( \mathbf{M}(\dot{\theta}) = \text{diag}(M_0(x_0), M_{AP}(\dot{q})) \), \( M_{AP}(\dot{q}) = P_{AP} \text{diag}(q_1, \ldots, q_m) \), \( M_{AP}(\dot{q}) = P_{AP} \text{diag}(q_1, \ldots, q_m) \), \( C_{AP}(\dot{q}, \dot{q}) = P_{AP} \text{diag}(C_{AP}(\dot{q}, \dot{q})) \), \( g(\dot{\theta}) = g_{AP}(\dot{q}) \), and \( P_{AP} = P_{AP}(q) \).

Considering the orthogonal decomposition of the projection of
\[
\dot{\mathbf{M}}(\dot{\theta}) \ddot{\theta} + \mathbf{C}(\dot{\theta}, \dot{\theta}) \dot{\theta} + \mathbf{g}(\dot{\theta}) + \tau_d = \tau_v + \mathbf{A}^T(x_0) \dot{\mathbf{h}}_{OM}, \tag{19}
\]
where \( \tau_v \) is an auxiliary control input
\[
\tau_v = \begin{bmatrix} \mathbf{A}^T h_{OM} \\ J_{AP}^T(x_0) \dot{\mathbf{h}}_{OM} \end{bmatrix},
\]
and \( \mathbf{A}(x_0) = [\mathbf{A}^T_1(x_0) \mathbf{A}^T_2(x_0) \mathbf{A}^T_3(x_0)] \) is a Jacobian matrix. If the auxiliary control input is partitioned into three vectors, \( \tau_v = \mathbf{A}^T_1 h_{OM}, \tau_v = \mathbf{A}^T_2 h_{OM}, \) and \( \tau_v = \mathbf{A}^T_3 h_{OM} \), the applied torque in the active joints can be computed as
\[
\tau_1 = \tau_2 = \tau_3 = \begin{bmatrix} J_{AP}^T(x_0) \dot{\mathbf{h}}_{OM} \\ J_{AP}^T(x_0) \dot{\mathbf{h}}_{OM} \end{bmatrix} \mathbf{A}^T(x_0) h_{OM}, \tag{20}
\]
\[
\tau_v = \tau_\mathbf{A}^T(x_0) h_{OM}.
\]

The update and control laws for underactuated cooperative manipulators can now be defined as
\[
\dot{\Theta} = \text{Proj}_{Z}^{-1} \tilde{\Theta} = \begin{bmatrix} \mathbf{A}^T h_{OM} + \mathbf{A}^T \dot{\mathbf{h}}_{OM} \end{bmatrix}, \tag{21}
\]
where \( \hat{\mathbf{h}}_{OM} \) is the full rank matrix that projects the null space of \( \mathbf{A}^T \), that is, \( \mathbf{A}^T \hat{\mathbf{h}}_{OM} = \mathbf{0} \). Hence, the \( \eta(m-1) \)-dimensional vector \( \lambda_{\mathbf{h}_{OM}} \) is a variable to be controlled.

The squeeze force error term, \( \rho(n) \mathbf{A}^T(x_0) \hat{\mathbf{h}}_{OM} \), can be described as
\[
\tau_{\mathbf{h}_{OM}} = \begin{bmatrix} \mathbf{A}^T \\ \mathbf{A}^T \end{bmatrix} \begin{bmatrix} \mathbf{A}^T(x_0) \hat{\mathbf{h}}_{OM} \\ J_{AP}^T(x_0) \end{bmatrix} \mathbf{A}^T \lambda_{\mathbf{h}_{OM}}, \tag{22}
\]
where \( \tau_{\mathbf{h}_{OM}}, \tau_{\mathbf{h}_{OM}}, \tau_\mathbf{h}_{OM} \) are the contributions of the squeeze force error in the components of the auxiliary control input \( \tau_v \). By imposing that \( \tau_{\mathbf{h}_{OM}} = 0 \), since it is assumed that no actuation occurs at passive joints, \( \eta \) constraints are created in the components of \( \lambda_{\mathbf{h}_{OM}} \).

Considering the kinematic constraints (17), the dynamic equation of the underactuated cooperative manipulator is given by
\[
\dot{\mathbf{M}}(\dot{\Theta}) \ddot{\Theta} + \mathbf{C}(\dot{\Theta}, \dot{\Theta}) \dot{\Theta} + \mathbf{g}(\dot{\Theta}) + \tau_d = \tau_v + \mathbf{A}^T \dot{\mathbf{h}}_{OM}.
\]

5. Experimental results

The underactuated cooperative manipulator of Fig. 1 will be controlled based on the neural network-based \( \mathcal{H}_\infty \) controllers developed in this paper. It is composed of two planar underactuated manipulators UArm II (Underactuated Arm II). All joints of this three-link robot can be configured as active or passive ones. The kinematic and dynamic parameters can be found in (Siqueira and Terra (2004)). The object parameters are presented in Table 1. Fig. 2 shows the workspace and the coordinate system for the cooperative manipulator.

The desired trajectory is an arc of the circle, centered at \( [0.24 \text{ m}] \) and with radius \( R = 0.26 \text{ m} \). The arc is from \( x_0(0) = [0.1 \text{ m} \ 0.3 \text{ m}]^T \) to \( x(T) = [0.38 \text{ m} \ 0.3 \text{ m}]^T \), where \( T = 3 \text{ s} \) is the trajectory duration time, see (Fig. 2). The reference trajectory for the x-axis is a fifth degree polynomial, and for the y-axis it is defined by the arc of the circle. External disturbances were introduced to verify the robustness of the
proposed controllers:

\[
\tau_{d_1} = \begin{bmatrix}
0.01e^{-t/0.5} \sin(2\pi t) \\
-0.01e^{-t/0.5} \sin(2.5\pi t) \\
-0.01e^{-t/0.5} \sin(3\pi t)
\end{bmatrix}
\] and

\[
\tau_{d_2} = \begin{bmatrix}
0.02e^{-t/0.5} \sin(2\pi t) \\
0.02e^{-t/0.5} \sin(2.5\pi t) \\
0.01e^{-t/0.5} \sin(3\pi t)
\end{bmatrix}.
\]

These disturbances, shown in Fig. 3, were added to the applied torques in the joints to guarantee the repeatability of the experiment.

5.1. Fully actuated configuration

For the neural network-based \( H_\infty \) control described in Section 3, the following values are used: \( K = \text{diag}[3.42l_1, 0.38l_3, A = 0.8l_3, A_f = 0.5l_3, \rho = 0.4, \eta = 1, \text{and } S = 50] \). The desired values for the squeeze force are \( h_{ax}^d = [0 \ 0 \ 0]^T \).

Table 1: Object parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mass</td>
<td>m_o = 1.45 kg</td>
</tr>
<tr>
<td>Length</td>
<td>l_o = 0.120 m</td>
</tr>
<tr>
<td>Center of mass</td>
<td>a_o = 0.060 m</td>
</tr>
<tr>
<td>Inertia</td>
<td>I_o = 0.0026 kg m^2</td>
</tr>
</tbody>
</table>

For the neural network computation, the following auxiliary variable is defined:

\[
xx = \sum_{i=1}^{n}(x_o) + (q_o) + (\dot{q}_o).
\] (23)

The matrix \( \Xi \) can be computed as \( \Xi = \text{diag}[\xi_1^T, \xi_2^T, \ldots, \xi_7^T] \) with \( \xi_k = \left[ \xi_{k1}, \ldots, \xi_{k7} \right]^T \), \( \xi_{k1} = (e^{xx-m_k} - e^{-xx-m_k})/(e^{xx-m_k} + e^{-xx-m_k}) \), where \( m_k \) assumes the values \(-1.5, -1, -0.5, 0, 0.5, 1, 1.5 \), for \( i = 1, \ldots, 7 \), respectively. Note that, with these definitions, seven neurons in the hidden layer are selected for the neural networks with the weights \( w_{ij} \), assuming the value 1. The network parameters \( \theta \) are defined as \( \theta = [\theta_1, \ldots, \theta_5]^T \), with \( \theta_k = [\theta_{k1}, \theta_{k2}, \ldots, \theta_{k7}]^T \).

To apply the VSC algorithm, it is assumed that the approximation error is bounded by the state-dependent function \( k(x) \) defined as \( k(x) = 2 \sqrt{x_1^2 + x_2^2} \). The experimental results, Cartesian coordinates and orientation of the object, are shown in Fig. 4. Figs. 5 and 6 show the joint positions and torques of the arms A and B.

Three performance indexes are used to compare the neural network-based \( H_\infty \) controller with the nonlinear \( H_\infty \) controllers presented in Siqueira and Terra (2007b): the \( L_2 \) norm of the state vector,

\[
L_2(x) = \left( \int_0^T \| x(t) \|^2_2 \right)^{1/2},
\]

where \( \| \cdot \|_2 \) is the Euclidean norm; the sum of the applied torques by the \( i \)-th joint for both manipulators,

\[
E[\tau] = \sum_{i=1}^{n} \left( \sum_{j=1}^{k} (\tau_{ij}(t) \right) dt
\]

and the sum of the squeeze forces,

\[
E[h_{ax}] = \sum_{i=1}^{nm} \left( \int_0^{t_e} |h_{ax}(t) dt \right) .
\]

where \( t_e \) is the spent time for the object to reach the desired position. The results presented in Tables 2 and 3 are the average of the respective experiments which were run five times.

Table 2 shows the values of \( L_2(x) \), \( E[\tau] \) and \( E[h_{ax}] \) computed with the results obtained from the implementation of the neural network-based \( H_\infty \) controller and the nonlinear \( H_\infty \) controllers presented in Siqueira and Terra (2007b), considering the fully actuated configuration.

Note that the neural network-based \( H_\infty \) controller presented the lowest values of trajectory tracking error, \( L_2(x) \), and spent energy, represented by the sum of the applied torque \( E[\tau] \). The sum
of the squeeze forces, $E_{h_{0}\delta}$, is equivalent to the values of the nonlinear $H_{\infty}$ controllers via quasi-LPV and via game theory.

5.2. Underactuated configuration

In this section, it is considered that joint 1 of arm A is passive (Fig. 2). In this case, only two components of the squeeze force can be controlled independently ($n_s = n(m - 1) - n_p = 3(2 - 1) - 1 = 2$) (Tinós & Terra, 2006). It is assumed that the component of the squeeze force referring to the momentum applied to the object will not be controlled. The desired values for the squeeze force are $\lambda_{d_{\infty}} = [0 \ 0]^T$.

The values used for the implementation of the neural network-based $H_{\infty}$ control are: $K = \text{diag}[3.42I_3, 0.38I_6]$, $A = 1.5I_3$, $A_f = 0.4I_3$, $\rho = 0.2$, $\eta = 1$, and $S = 50$. The experimental results are shown in Figs. 7–9. The performance indexes are presented in Table 3. Note that, again, the neural network-based $H_{\infty}$ controller presented the lowest values of trajectory tracking error and total spent energy. The sum of the squeeze forces is practically the same for all controllers.

<table>
<thead>
<tr>
<th>$H_{\infty}$ controller</th>
<th>$\mathcal{L}_2[\mathcal{E}]$</th>
<th>$E_t$ (N m s)</th>
<th>$E_{h_{0}\delta}$ (Ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Neural network-based</td>
<td>0.0267</td>
<td>1.09</td>
<td>0.9253</td>
</tr>
<tr>
<td>Quasi-LPV</td>
<td>0.0514</td>
<td>1.95</td>
<td>1.02</td>
</tr>
<tr>
<td>Game theory</td>
<td>0.0617</td>
<td>2.30</td>
<td>1.11</td>
</tr>
</tbody>
</table>

Table 2

Fully actuated configuration
The same experiment was also implemented using the no robust hybrid position/force controller for underactuated manipulators proposed in Tinós and Terra (2006). The performance indexes are given by: $L_2[e] = 0.0651$, $E[r] = 2.95$, and $E[h_{os}] = 2.03$. In comparison with the neural network-based controller (Table 3), the performance indexes $L_2[e]$, $E[r]$, and $E[h_{os}]$ are poorer (35%, 25% and 30%, respectively). It can also be observed that, the value of $L_2[e]$ is practically the same, whether it is obtained with the nonlinear $H_{\infty}$ controllers via quasi-LPV or via game theory. However, the values of $E[r]$ and $E[h_{os}]$ are poorer. These comparative results remain valid when one joint is not actuated, for different initial and final positions.

### 6. Conclusions

In this work, a neural network-based $H_{\infty}$ controller was developed for fully actuated and underactuated cooperative manipulators. The new feature of this controller is the use of neural networks to approximate only the uncertain parameters of the robot, whose mathematical model is well established. The experimental results obtained in an actual underactuated cooperative manipulator show that position and squeeze force errors can be robustly controlled, even when one joint is not actuated. They also show that in comparison with well-established robust control techniques, this approach provides superior performance.

### Table 3

<table>
<thead>
<tr>
<th>$H_{\infty}$ controller</th>
<th>$L_2[e]$</th>
<th>$E[r]$ (N m s)</th>
<th>$E[h_{os}]$ (N s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Neural network-based</td>
<td>0.0462</td>
<td>2.36</td>
<td>1.56</td>
</tr>
<tr>
<td>Quasi-LPV</td>
<td>0.0561</td>
<td>2.70</td>
<td>1.61</td>
</tr>
<tr>
<td>Game theory</td>
<td>0.0574</td>
<td>2.93</td>
<td>1.65</td>
</tr>
</tbody>
</table>

**Fig. 7.** Coordinates X, Y and orientation, $\phi$, for an underactuated configuration.

**Fig. 8.** Joint positions, arms A and B, for an underactuated configuration.

**Fig. 9.** Torques, arms A and B, for an underactuated configuration.
Appendix A. Proof of theorem

Consider the Lyapunov function
\[ V = \frac{1}{2} s^T \dot{S} \mathbf{x} + \frac{1}{2} \rho \eta \sigma^T \mathbf{J}_d^T e + \frac{1}{2} \mathbf{Q}^T \hat{S} \mathbf{Q}. \]

The time derivative of \( V \) along the error dynamics (9) and control law (16) is
\[ \dot{V} = s^T \dot{S} \mathbf{x} + \frac{1}{2} \rho \eta \sigma^T \mathbf{J}_d^T e + \frac{1}{2} \mathbf{Q}^T \hat{S} \dot{\mathbf{Q}}. \]

From the definition of the update law (15), where the projection algorithm is used, it can be shown that
\[ \dot{\theta}^T S \mathbf{x} + s^T \hat{S} \mathbf{x} < 0 \]
and \( \theta(t) \in \Omega_d \) for all \( t \geq 0 \) if \( \theta(0) \in \Omega_0 \), with \( \Omega_0 = \{ \theta : \theta^T \theta \leq M \} \). Taking into account the control law \( \tau_c \) and the assumption (ii), it can be guaranteed that
\[ s^T \dot{\mathbf{x}} = s^T (M_0(\mathbf{x}_o) - \mathbf{D} \dot{\mathbf{x}}), \]
\[ = s^T \mathbf{D} \mathbf{x}_o + s^T \mathbf{D} \dot{\mathbf{x}}. \]

From (28), \( V \) is convergent, which implies \( s \in L_\infty \), and then, \( \varepsilon_o \), \( \varepsilon_d \), and \( \varepsilon_y \in L_\infty \). The squeeze force error can be expressed as function of the neural network approximation parameters \( (\Delta \mathbf{M}, \Delta \mathbf{A}, \Delta \mathbf{G}) \)
\[ \eta \mathbf{R}(\mathbf{x}_o) M^{-1}(\mathbf{x}_d) \mathbf{R}(\mathbf{x}_o) + \rho / (\eta^2) \mathbf{M} \mathbf{E} \mathbf{E}^T \mathbf{R}(\mathbf{x}_o) \mathbf{h}_{\mathbf{h}_o} = z - \eta \mathbf{R}(\mathbf{x}_o) M^{-1}(\mathbf{x}_d) \mathbf{M} \mathbf{E} e + \mathbf{R}(\mathbf{x}_o) \dot{\mathbf{b}}(\mathbf{x}_o) \mathbf{x}_o. \]

Since all terms on the right-hand side of (31) are bounded, a proper value of \( \rho \) assures \( \mathbf{h}_o \) is bounded, and then, \( \varepsilon_o \), \( \varepsilon_d \), and \( \varepsilon_y \in L_\infty \). If a square-integrable disturbance is assumed, i.e., \( \tau_d \in L_2 \), then \( s \in L_2 \) by integrating (28). By Barbalat’s lemma, \( \lim_{t \to \infty} \mathbf{x}_o(t) = 0 \), since \( s \in L_2 \) and \( \dot{s} \in L_\infty \). Hence, \( \lim_{t \to \infty} \varepsilon_o = 0 \).

References


Please cite this article as: Siqueira, A. A. G., & Terra, M. H. Neural network-based \( \mathcal{H}_\infty \) control for fully actuated and underactuated cooperative manipulators. Control Engineering Practice (2008). doi:10.1016/j.conengprac.2008.09.007