Neural PID controller to improve hierarchical multitask scheme for redundant cooperative manipulators

Arturo A. Perez[2], Jesus Hernandez-Barragan[2], Javier Gomez-Avila[2], Carlos Lopez-Franco[1], Nancy Arana-Daniel[1]

[1] Departamento de Ciencias Computacionales, Centro Universitario de Ciencias Exactas e Ingenierías, Universidad de Guadalajara, Blvd. Marcelino García Barragán 1421, Guadalajara C.P. 44430, Jalisco, Mexico;
[2]Departamento de Innovación Basada en la Información y el Conocimiento, Centro Universitario de Ciencias Exactas e Ingenierías, Universidad de Guadalajara, Blvd. Marcelino García Barragán 1421, Guadalajara C.P. 44430, Jalisco, Mexico;

Abstract. The hierarchical multitasking scheme is a useful approach for robotic systems with multiple objectives, where each objective has a lower priority than the previous one. However, some tasks can be complex enough to require significant effort for fine-tuning. In this paper, we propose the implementation of a neural incremental proportional-integrative-derivative (NI-PID) approach to control multiple tasks within a hierarchical multitasking scheme for a dual-arm robotic system. The NI-PID adapts its weights online, significantly reducing the effort required for tunning gains. The effectiveness of the proposed approach is validated through simulations on a dual-arm system modeled with the Baxter arm, featuring 7 degrees of freedom (7DOF).

1 Introduction

The cooperation and collaboration of robotic manipulators remain a highly studied area, the proper functioning of two or more manipulators working together has a positive impact on various fields, such as industry, aeronautics, exploration systems, and more, reducing costs, increasing production efficiency, and enhancing automation capabilities while carrying out tasks classified as risky for humans. Both terms might be considered synonyms; however, when it refers to the topic of controlling robotic manipulators, they have important differences [1]. Cooperation refers to how two or more manipulators fulfill independent specific tasks that contribute to achieving a final common objective. Each manipulator's task is independent of the others, and while it can significantly impact the final objective, it does not affect the task of each manipulator. On the other hand, Collaboration suggests that two or more manipulators share a common objective, where each one performs a specific task that highly depends on the performance of the other manipulator [2, 3]. Coordination in cooperation and collaboration tasks suggests that the manipulators physically interact with each other, meaning their dynamics must be related. Conversely, in non-coordinated scenarios, robotic manipulators can be analyzed individually, as their dynamics do not

https://doi.org/10.61728/AE20255275

interact in any way [4]. In this work, we implement a hierarchical multitasking scheme for a dual-arm system in coordinating cooperative manipulation tasks.

Kinematic redundancy is important to consider when working with robotic manipulators, it refers to the robot having more degrees of freedom than necessary to perform a task. It is well known that six degrees of freedom are sufficient to execute most tasks in the Cartesian workspace. However, a higher number of degrees of freedom yield better results when performing more complex tasks or tasks requiring greater precision [5]. A robotic manipulator can be controlled to successfully execute two or more tasks depending on their complexity and the degrees of freedom it possesses; this concept is known as multitask scheme. Furthermore, if the tasks need to be executed with a certain level of priority, the concept is named as hierarchical multitask scheme. In the last one, the robotic manipulator has a control action that seeks to satisfy multiple objectives simultaneously, such as trajectory tracking, obstacle avoidance, and coordination, among others, where each task can be weighted based on its desired priority, thus establishing an order and hierarchy during execution [6].

Classical control methods often get good results when applied to robotic manipulators, but they come with certain challenges, such as the need for recalibration over time, lack of adaptability, and difficulty in adjusting gains for specific work environments. One of the proposed solutions to these challenges, which has gained popularity in recent years, is neural controllers. These controllers can modify gains either online or offline, improving results and enhancing the manipulator's adaptability to noise or external system elements [7]. The contribution of this paper is to propose a better solution when handling hierarchical multitask schemes for redundant robots.

In this work, we propose improving a hierarchical multitask scheme for coordinate cooperative manipulators. To improve adaptability against external conditions and avoid extra efforts tunning parameters, the classic controller for coordinate manipulation is going to be replaced for a NI-PID. The case of study consists of a dual-arm system of two Baxter 7DOF manipulators. The master manipulator must execute a secondary task which consists of following a trajectory, while both robots (master and slave) are fulfilling the primary task of coordination. The kinematics for both manipulators are analyzed together using a relative Jacobian. This problematic can be applied to a wide range of real-life situations, where the tasks and their priority can be modified to achieve different objectives.

The paper is organized as follows: Next section introduces the Hierarchical multitasking scheme along with the relative Jacobian. Section 3 addresses the NI-PID controller and its adaptations rules. Section 4 presented the proposed approach. The simulation results are given in section 5. Finally, the conclusions are provided in section 6.

2 Hierarchical multitasking scheme

Hierarchical multitasking scheme allows multiple tasks to be executed in a structured order, where each task does not interfere with the execution of the preceding one [8]. This structure continues recursively, ensuring that task N does not affects task $N-1$, maintaining system stability and efficiency. In this paper, we consider implementing the hierarchical multitasking scheme for a dual-arm system for coordinate cooperation.

The dual-arm system scheme for coordinate cooperation is represented in Fig. 1. The vector $\vec{q}_A$ contains the joint configuration for manipulator A, such that $\vec{q}_A = [\theta_1^A, \theta_2^A, \theta_3^A, \theta_4^A, \theta_5^A, \theta_6^A, \theta_7^A]^T$; moreover, for manipulator B we have $\vec{q}_B = [\theta_1^B, \theta_2^B, \theta_3^B, \theta_4^B, \theta_5^B, \theta_6^B, \theta_7^B]^T$. Both manipulator joints are contained on $\vec{q} = [q_A^T, q_B^T]^T$. Matrixes J_A and J_B expressed the Jacobian matrix in the world frame for manipulator A and B, respectively.

For the coordination objective on a multitask scheme, the analysis of the robotic manipulators is done together, as shown in Fig. 1, it is defined four important reference frames: the base of manipulator A $\{1\}$, its end-effector $\{2\}$, the base of manipulator B $\{4\}$, and its end-effector $\{3\}$.

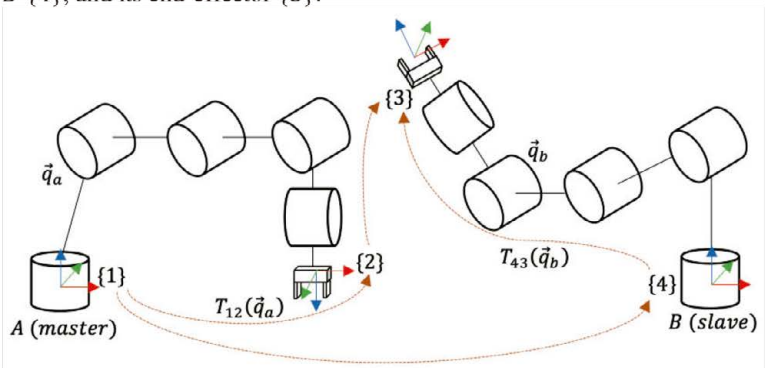

Fig. 1. Dual-arm system for coordinate cooperation.

In Fig. 1, T_{ij} is a homogenous transformation matrix that transforms frame $\{i\}$ to frame $\{j\}$. This matrix is defined as

$$T_{ij} = \begin{bmatrix} R_{ij} & \vec{t}_{ij} \\ \vec{0} & 1 \end{bmatrix} \tag{1}$$

where R_{ij} is a 3x3 rotation matrix, and t_{ij} is a translation vector.

Following the literature [1, 4], a matrix that relates end-effector for manipulator A with end-effector for manipulator B relatively to their bases frame of reference can be defined as relative Jacobian:

$$J_R(\vec{q}) = [-\psi_{23}\Omega_{21}J_A \quad \Omega_{24}J_B] \tag{2}$$

where:

$$\psi_{ij} = \begin{bmatrix} I & -S(\vec{t}_{ij}) \\ \vec{0} & I \end{bmatrix} \qquad \Omega_{ij} = \begin{bmatrix} R_{ij} & 0 \\ 0 & R_{ij} \end{bmatrix} \tag{3}$$

and $S(\cdot)$ is an antisymmetric matrix generated by vector $\vec{t}_{ij}$.

The classic hierarchical multitask scheme for two tasks, where the primary is coordination using the relative Jacobian is:

$$\dot{\vec{q}} = J_R^+(\dot{\vec{x}}_{R_d} + K_1\vec{e}_R) + (I - J_R^+J_R)J_k^+(\dot{\vec{x}}_{k_d} + K_2\vec{e}_k) \tag{4}$$

where J_R^+ is the Moore-Penrose pseudo-inverse of the relative Jacobian, $\dot{\vec{x}}_{R_d}$ is the desired relative end-effector motion vector, K_1 and K_2 are gains matrix that define the convergence, $\vec{e}_R$ is the relative position and rotation error, the term $(I - J_R^+J_R)$ is the null space projection of the primary task, J_k is a Jacobian matrix designed for the secondary task, $\dot{\vec{x}}_{k_d}$ is the desired motion vector for secondary task, and $\vec{e}_k$ is the error defined for secondary task.

Gains K_1 and K_2 are tunned manually to achieve best performance. However, in this work we proposed to use NI-PID approach to adapt them online.

3 Neural incremental PID approach

The structure for the NI-PID is illustrated on Fig. 2, and the control action given by the NI-PID is defined by [9]:

$$u(t) = u(t-1) + \Delta u(t) = u(t-1) + K_{PID}\sum_{m=1}^{3}\omega_m(t)y_m(t) \tag{5}$$

where K_{PID} is a scalar gain, ω_m are the normalized weights, and $y_m(t)$ are the neuron inputs that represent the PID errors:

$$\begin{aligned} y_1(t) &= e_{PID}(t) \\ y_2(t) &= e_{PID}(t) - e_{PID}(t-1) \\ y_3(t) &= e_{PID}(t) - 2e_{PID}(t-1) + e_{PID}(t-2) \end{aligned} \tag{6}$$

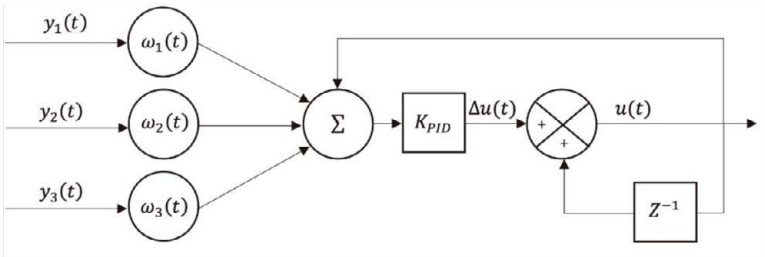

Fig. 2. Structure for NI-PID.

The input y_1 represents integral term, y_2 is the proportional, and y_3 the derivative, while e_{PID} is the error that wants to be minimized with this controller. To prevent that weight grows without bounds, we normalize them.

Normalized weights are obtained with:

$$\omega_m(t) = \frac{w_m(t)}{\sum_{n=1}^{3}|w_n(t)|} \tag{7}$$

where w_m are the regular weights for the neuron.

Update weights rule is:

$$\begin{aligned} w_1(t) &= w_1(t-1) + \eta_I e_{PID}(t)u(t)y_1(t) \\ w_2(t) &= w_2(t-1) + \eta_P e_{PID}(t)u(t)y_2(t) \\ w_3(t) &= w_3(t-1) + \eta_D e_{PID}(t)u(t)y_3(t) \end{aligned} \tag{8}$$

where the scalar values for η_I, η_P, η_D are the learning rates for the proportional, integral and derivative terms [9].

In summary, given an error $e_{PID}(t)$, the NI-PID computes the control law $u(t)$, which is $u(t)$ =NI-PID($e_{PID}(t)$).

4 Proposed approach

The hierarchical multitask scheme (4) for coordinate manipulation is considered for two tasks. We consider a primary task error $\vec{e}_R = |\vec{e}_{23t}^T, \vec{e}_{23r}^T|^T$, where $\vec{e}_{23t}$ is the translational error obtained from the difference between the relative translation $\vec{t}_{23}$ and the desired relative translation $\vec{t}_{23}^*$, which is $\vec{e}_{23t} = \vec{t}_{23} - \vec{t}_{23}^*$. The $\vec{e}_{23r}$ is the rotational error obtained from the cross product of the relative rotation R_{23} and the desired relative translation R_{23}^* divided by half:

$$\vec{e}_{23r} = \frac{1}{2}\left(R_{23}\begin{bmatrix}1\\0\\0\end{bmatrix} \times R_{23}^*\begin{bmatrix}1\\0\\0\end{bmatrix} + R_{23}\begin{bmatrix}0\\1\\0\end{bmatrix} \times R_{23}^*\begin{bmatrix}0\\1\\0\end{bmatrix} + R_{23}\begin{bmatrix}0\\0\\1\end{bmatrix} \times R_{23}^*\begin{bmatrix}0\\0\\1\end{bmatrix} \right) \tag{10}$$

Similarly, the error for neural network on following trajectory task is $\vec{e}_k = [\vec{e}_t^T, \vec{e}_r^T]^T$, where the translational error is $\vec{e}_t = [\vec{t} - \vec{t}^*]$, and the rotational error is

$$\vec{e}_r = \frac{1}{2}\left(R_{12}\begin{bmatrix}1\\0\\0\end{bmatrix} \times R^* \begin{bmatrix}1\\0\\0\end{bmatrix} + R_{12}\begin{bmatrix}0\\1\\0\end{bmatrix} \times R^* \begin{bmatrix}0\\1\\0\end{bmatrix} + R_{12}\begin{bmatrix}0\\0\\1\end{bmatrix} \times R^* \begin{bmatrix}0\\0\\1\end{bmatrix} \right) \tag{11}$$

being $\vec{t}^*$ and $\vec{R}^*$ the translation and rotation desired for the end-effector of manipulator A.

Errors $\vec{e}_R$ and $\vec{e}_k$ are 6x1 vector that contains position and rotations errors for primary and secondary tasks, respectively. Λ NI-PID controller is applied independently for each error. Then, we have six NI-PID controllers to compute control signals for primary task $\vec{u}_R$ based on $\vec{e}_R$, and other six NI-PID controllers to compute control signals for secondary task $\vec{u}_k$ based on $\vec{e}_k$. Eq. (4) is rewritten to

$$\dot{\vec{q}} = J_R^+\left(\dot{\vec{x}}_{R_d} + \vec{u}_R\right) + (I - J_R^+ J_R) J_k^+\left(\dot{\vec{x}}_{k_d} + \vec{u}_k\right) \tag{9}$$

where $J_k = [J_A, 0]$.

5 Simulation results

The proposed approach is validated through simulations. We consider a dual-arm system composed of two Baxter arm. The Denavit-Hartenberg (DH) parameters are shown in Table 1, since both manipulators are going to have the same configuration for this paper purposes, the individual kinematics for manipulator A and manipulator B are equal [10].

Table 1. Baxter arm DH parameters.

Joint	A[m]	α[rad]	d[m]	θ[rad]
1	0.069	$-\pi/2$	0.270	θ_1
2	0	$\pi/2$	0	θ_2
3	0.069	$-\pi/2$	0.364	θ_3
4	0	$\pi/2$	0	θ_4
5	0.01	$-\pi/2$	0.374	θ_5
6	0	$\pi/2$	0	θ_6
7	0	0	0.28	θ_7

As shown in Fig. 3, both manipulators are in the exact same start position, with a offset of 0.5 meters on y axis.

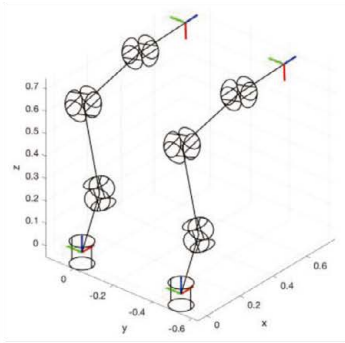

Fig. 3. Dual-arm system for simulations base on two 7DOF baxter arms.

The first experiment can be seen on Fig. 4 and Fig. 5, where both controllers were tested trying to have a coordination task for $\vec{t}_{23d} = [0 \quad -0.5 \quad 0.2]^T$, while the second task is the master manipulator trying to get to $\vec{t}_d = [0.7 \quad 0 \quad 0.5]^T$.

Fig. 4 shows the classic controller results, as can be observed, the primary task is performed, then the secondary is tried to complete. These results probably could be improved after hours of gain tunning.

Fig. 5 shows how the NI-PID approach has a similar behavior on the primary task, but results for secondary task are better, even when the weights are initialized with random numbers between 0 and 1.

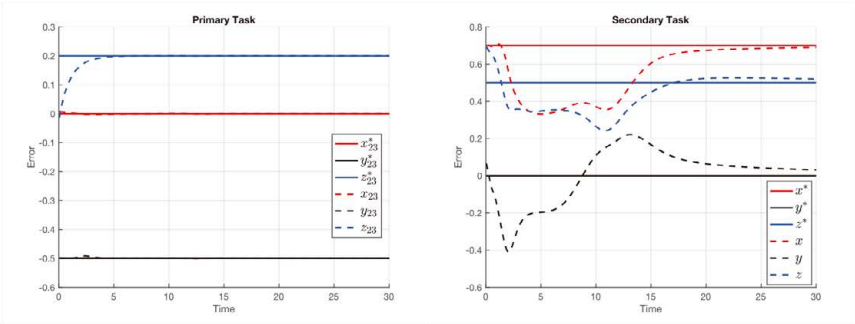

Fig. 4. Classical approach while primary and secondary task are constants.

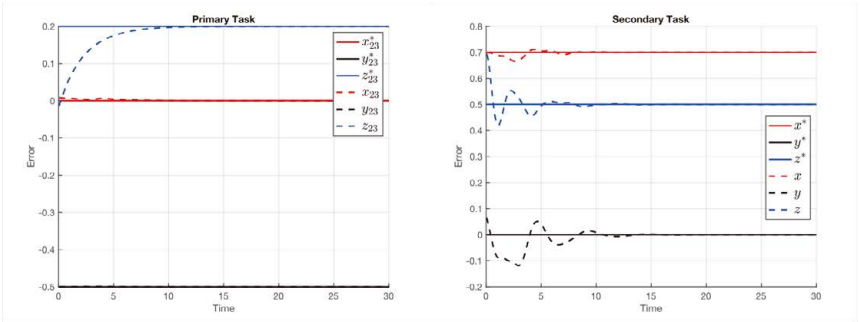

Fig. 5. NI-PID approach while primary and secondary task are constants.

The Fig. 6 and Fig. 7 show both approaches against task that are time variable, which is the second experiment. Primary task is trying to follow a sinusoidal signal of 0.1 amplitude and 0.3 frequency, while the secondary task tries to follow a sinusoidal trajectory of 0.1 amplitude and 0.1 frequency.

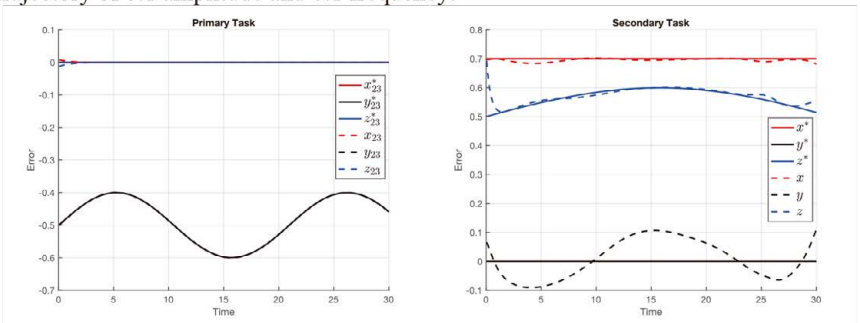

Fig. 6. Classic approach while primary and secondary task are time variant trajectories.

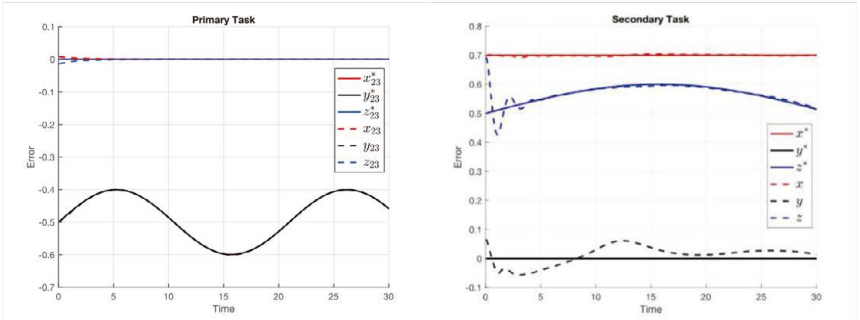

Fig. 7. NI-PID approach while primary and secondary task are time variant trajectories.

6 Conclusions

In this paper, we proposed the implementation of a NI-PID control for coordinate manipulation based on a hierarchical multitasking scheme. Simulation results on a dual-arm system prove that NI-PID adapts the controller weights online where trajectory errors are significantly reduced without, rather than using fixed weights. The use of NI-PID controller avoids the effort required for tunning control gains in classic approach. The use of NI-PID control can be extended for coordinate manipulation in dual mobile manipulator systems, where kinematics modeling and control are more challenging, and the gains tunning can be difficult and tedious.

References

1. Ortenzi, D., Muthusamy, R., Freddi, A., Monteriù, A., & Kyrki, V. (2018). Dual-arm cooperative manipulation under joint limit constraints. Robotics and Autonomous Systems, 99, 110–120.
2. Kolbeinsson, A., Lagerstedt, E., & Lindblom, J. (2019). Foundation for a classification of collaboration levels for human-robot cooperation in manufacturing. Production and Manufacturing Research: An Open Access Journal, 7(1), 448–471.
3. Ghorbanpour, A. (2023). Cooperative robot manipulators dynamical modeling and control: An overview. Dynamics, 3(4), 820–854.
4. Sun, D., & Liao, Q. (2024). Real-time coordination of multiple robotic arms with reactive trajectory modulation. IEEE Transactions on Robotics, 41(1), 200–219.
5. Hernandez-Barragan, J., Lopez-Franco, C., Arana-Daniel, N., & Alanis, A. Y. (2021). Inverse kinematics for cooperative mobile manipulators based on self-adaptive differential evolution. PeerJ Computer Science, 7, e419.
6. Jamisola, R., & Roberts, R. (2015). A More Compact Expression of Relative Jacobian Based on Individual Manipulator Jacobians. Robotics and Autonomous Systems, 63, 158–164.
7. Moran Armenta, M. A. (2022). Control de robots manipuladores usando redes neuronales. Instituto Politécnico Nacional, Centro de Investigación y Desarrollo de Tecnología Digital, México.
8. Freddi, A., Longhi, S., Monteriu, A., & Ortenzi, D. (2016). Redundancy analysis of cooperative dual-arm manipulators. International Journal of Advanced Robotic Systems, September-October, 1–14.
9. Miao, L., & Shu, Z. (2024). Neural incremental PID approach for hierarchical multitask schemes. Journal of Physics: Conference Series, 2816, 012031.
10. Williams, R. L. (2017). Baxter humanoid robot kinematics. Ohio University. Retrieved from: https://sites.ohio.edu/williams/html/PDF/BaxterKinematics.pdf.