# Robust PD-type Iterative Learning Control design for trajectory tracking of a Pelican robot

**Khouloud GUESMI[1,2], Imen SAIDI[1,2]**

[1] Université de Tunis El Manar, Ecole Nationale d'Ingénieurs de Tunis, Laboratoire de Recherche en Automatique, Tunis, Tunisia

[2] Université de Tunis, Ecole Nationale Supérieure d'Ingénieurs de Tunis, Tunis, Tunisia

guesmikhouloud92@gmail.com, imen.saidi@gmail.com

**Abstract:** This work focuses on the application of the Proportional-Derivative (PD)-type Iterative Learning Control (ILC) to the two-degree-of-freedom (2DOF) Pelican robot manipulator, particularly in the context of cyclic tasks, aiming to address the persistent challenges of trajectory tracking and steady-state error. To ensure that the system remains stable and that errors diminish over time, the control law is updated iteratively. The simulations carried out on the Pelican robot clearly demonstrate the effectiveness and robustness of this method. By taking past tracking errors into account, the control is gradually refined. This learning process not only enhances trajectory tracking precision but also allows the system to adapt its behavior in response to load changes and variations in operating conditions.

**Keywords:** PD-Type ILC, Trajectory Tracking, Pelican Robot, Steady-State.

## 1. Introduction

The control of robotic manipulators is currently one of the major concerns and preferred research areas in the field of automation. Indeed, the majority of the tasks assigned to robots are delicate and require very high precision under fast trajectories (Umar & Bakar,2014; Dehghani & Khodadadi, 2016; Amieur et al., 2022).

Over the past thirty years, researchers have explored a wide range of control strategies to improve the performance and reliability of the robotic manipulators. These efforts have led to the development of several well-known techniques, including Proportional-Integral-Derivative (PID) control (Cojuhari, 2011; Mohamad Yuden et al., 2017), computed torque control, which simplifies nonlinear robot dynamics to achieve accurate trajectory tracking (Bouakrif & Zasadzinski, 2016), as well as adaptive, variable structure, and fuzzy control approaches (Wei, 2018; Xu, Zhang & Mohammadzadeh, 2023).

In industrial contexts, robots often perform the same operation repeatedly. This repetition frequently results in similar tracking errors across cycles. Observing this behavior suggests that control systems should not treat each cycle independently. Instead, they can use the errors from previous iterations to improve their performance in the following ones. Control strategies based on learning make this possible by updating the command signals over time, which leads to a gradual reduction in tracking errors. This characterizes the Iterative Learning Control (ILC) (Bien & Xu, 1998; Bouakrif, Boukhetala & Boudjema, 2007; Shen & Wang, 2014).

The basic idea of the Iterative Learning Control was published in 1978 in Japan by (Uchiyama, 1978). However, since this article was published in Japanese, the work has not gained continuity. Prior to Uchiyama's work, Garden patented his results in 1971, which he had already realized in 1967 in the United States, titled "Learning Control of Actuators in Control Systems". This approach involves storing a control signal in the computer's memory and iteratively updating the control signal using the error between the actual response and the desired response. It was only in 1984, following the works of (Norouzi & Koch, 2020; Wang et al., 2022; Saidi & Touati, 2023), that ILC was explicitly defined.

ILC can be classified into two categories: autonomous ILC (off-line) and connected ILC (on-line). In the autonomous ILC, the control in the current iteration uses information from the previous iteration, while in the connected ILC, it uses information from both the current and the previous iterations simultaneously (Yu, Hou & Xu, 2018). Additionally, another type of ILC algorithm has

https://doi.org/10.33436/v35i3y202503

been developed using a positive definite Lyapunov sequence that decreases with iterations through an appropriate choice of control. Xu and Qu (1998) showed how the Iterative Learning Control (ILC) can be combined with a variable structure control using a Lyapunov-based approach. Their work extended to a wide variety of nonlinear systems. By applying the Lyapunov theory, they developed an ILC law integrated with robust control to address trajectory tracking in robotic manipulators. This method was later expanded to cover many other types of nonlinear systems. Building on this foundation and the previous control strategies, the present endeavour aims to develop an ILC-based control system that is both straightforward and effective. A PD controller because it is simple to implement, especially in industrial environments. The goal is to find a control method that works well but remains easy to use in real industrial settings.

The remainder of this paper is organized as follows. Section 2 introduces the Iterative Learning Control (ILC) framework and the PD-type ILC approach, which combines the fast response of a PD controller with the ability to learn from past errors. Section 3 presents the dynamic modeling of the Pelican robot manipulator. Section 4 discusses the simulation results, emphasizing the method's accuracy, consistency, adaptability to load variations, and robustness. Finally, Section 5 concludes the paper and outlines directions for future research.

## 2. Iterative learning control

In automation, a repetitive system refers to any setup that performs the same task repeatedly over a defined period, such as robotic manipulators, satellites, or industrial equipment. When designing controllers for these systems, the goal is to make sure they react well and don't keep making the same mistakes. Achieving a consistent trajectory accuracy can be challenging, since repeated reference inputs can cause the system to replicate errors from the previous cycles. Typically, the control system does not account for errors from the previous cycles, highlighting the importance of using all collected information to adjust the control for the next cycle, thereby optimizing the reference trajectory tracking (Meng & He, 2020; Liu et al., 2022; Wang et al., 2022). This need has led to the development of the Iterative Learning Control, an adaptive method that refines the control at each iteration to reduce the discrepancy between the desired trajectory and the system output from one iteration to the next (Bristow, Tharayil & Alleyne, 2006), as illustrated in Figure 1:

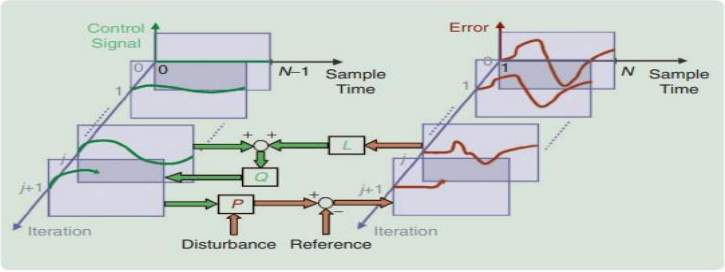


**Figure 1.** Standard ILC schematic diagram (Norouzi & Koch, 2020)

ILC fits within the broader framework of the closed-loop control, aiming to progressively improve the trajectory tracking accuracy. Unlike the traditional approaches that adjust the controller based on the system model, ILC iteratively utilizes historical data to optimize the trajectory tracking performance over time.

In this study, a Proportional-Derivative (PD) controller is employed instead of a Proportional-Integral-Derivative (PID) controller, as it provides a rapid response crucial for a precise trajectory tracking in repetitive robotic tasks while maintaining the system stability. The integral term of a PID, typically used to eliminate steady-state errors, is unnecessary here because the iterative learning process in PD-type ILC already corrects persistent errors across iterations. Moreover, using a PID increases the computational time since three parameters must be tuned and processed instead of two, which complicates the real-time implementation. Including the integral action could also introduce overshoot or oscillations, particularly in high-speed or sensitive applications such as the Pelican robot. By combining the PD control with the iterative learning, the

system achieves high accuracy and robustness without the additional computational burden of the PID tuning. This approach ensures an efficient, easy to implement control system suitable for industrial applications, while meeting the study's precision and stability requirements (Ouyang & Pipatpaibul, 2010; Riaz et al., 2023).

### 2.1. PD-Type Iterative Learning Control

Proportional-Derivative (PD) control proves especially effective for the robotic systems that need quick responses, although it does not ensure zero steady-state error (Chong et al., 2017). When following a path or managing their position, robots need to be able to react quickly. A PD controller uses two parts, one that reacts to the current error and one that reacts to how fast the error changes. This makes the robot respond quicker than if it only used one part. When you add Iterative Learning Control to the PD controller, the robot learns from the previous tries and adjusts its control little by little. Over time, this helps the robot do a better job and keep things steady. This enhanced control strategy is expressed by the following equations:

$$u_{k+1}(t) = u_k(t) + k_p\ e_k(t) + k_d\ \dot{e}_k(t) \tag{1}$$

where: $e_k(t)$ , $\dot{e}_k(t)$ , $k_p$ and $k_d$ are respectively the position tracking error vector, the velocity tracking errors vector , the proportional controller matrix and the derivative controller matrix.

## 3. Dynamic modeling of the Pelican robot

Consider the Pelican robot, consisting of $n$ degrees of freedom and $n$ joints. Its motion is described by the following dynamic equation (Bouakrif, Boukhetala & Boudjema, 2007):

$$M(q)\ddot{q} + C(q,\dot{q}) + C(q) = \tau \tag{2}$$

where $q(t)$ , $\dot{q}(t)$ and $\ddot{q}(t) \in \mathbb{R}^n$ are the position, velocity, and acceleration vectors respectively, $M(q) \in \mathbb{R}^{n\times n}$ denotes the inertia matrix, $C(q,\dot{q}) \in \mathbb{R}^{n\times n}$ is the Coriolis and centrifugal effects matrix, $G(q) \in \mathbb{R}^{n\times 1}$ represents the gravitational effect vector and $\tau \in \mathbb{R}^{n\times 1}$ is the vector of applied torques and forces on the actuators. The vectors $q_d(t)$ , $\dot{q}_d(t)$ and $\ddot{q}_d(t) \in \mathbb{R}^n$ represent the desired positions, velocities, and accelerations respectively. The dynamic equation (2) possesses the following properties (Bouakrif, Boukhetala & Boudjema, 2007):

Property 1: The matrix $M(q)$ is symmetric, positive definite, and bounded:

$$0 \le \lambda_{\min} I \le \| M(q) \| \le \lambda_{\max} I \tag{3}$$

where $\lambda_{\min}$ and $\lambda_{\max}$ are positive constants

Property 2: The norm of the matrix $C(q,\dot{q})$ is bounded as follows:

$$\| C(q,\dot{q}) \| \le \| \beta(\dot{q}) \| \tag{4}$$

where $\beta$ is a positive constant

Property 3: The norm of the vector $G(q)$ is bounded as follows:

$$\| G(q) \| \le \gamma \tag{5}$$

where $\gamma$ is a positive constant

To apply the control method, the authors introduce the experimental framework of the Pelican robot (Kelly, Santibáñez Dávila & Loría, 2005):

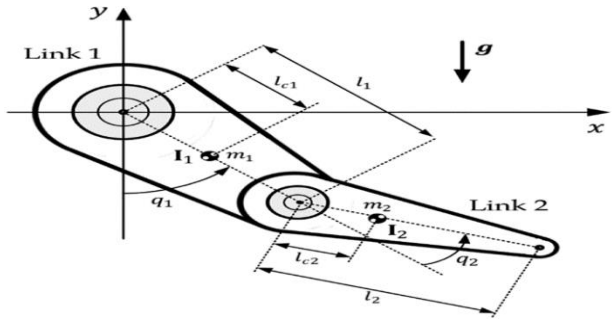


**Figure 2.** Schematic representation of the Pelican prototype (Kelly, Santibáñez Dávila & Loría, 2005)

The Pelican prototype consists of two rigid segments with lengths $l_1$ and $l_2$, and the respective masses $m_1$ and $m_2$. The robot's motion occurs in the $(x-y)$ plane, as illustrated in Figure 2. The distances from the rotation axes to the centers of mass are denoted $l_{c1}$ and $l_{c2}$ for segments 1 and 2, respectively. The moments of inertia of the segments about the axes passing through their respective centers of mass and parallel to the x-axis are designated as $I_1$ and $I_2$. Table 1 presents the detailed parameters of the Pelican robot (Kelly, Santibáñez Dávila & Loría, 2005):

**Table 1.** Pelican Prototype parameters

| Parameter | Designation | Value | Unit |
|---|---|---|---|
| Mass of Link 1 | $m_1$ | 6.5225 | kg |
| Mass of Link 2 | $m_2$ | 2.0458 | kg |
| Length of Link 1 | $l_1$ | 0.26 | m |
| Length of Link 2 | $l_2$ | 0.26 | m |
| Gravity | g | 9.81 | $m/s^2$ |
| Distance to Center of Mass 1 | $l_{c1}$ | 0.0983 | m |
| Distance to Center of Mass 2 | $l_{c2}$ | 0.0229 | m |
| Moment of Inertia at Center of Mass $m_1$ | $I_1$ | 0.1213 | $kg/m^2$ |
| Moment of Inertia at Center of Mass $m_2$ | $I_2$ | 0.1213 | $kg/m^2$ |

The dynamic model of the Pelican prototype can be calculated using the Lagrangian formalism. It is represented by the following equation (Sciavicco & Siciliano, 2000):

$$\tau_i = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} \tag{6}$$

$L = E_c - E_p$ is the Lagrangian function of the robot, $E_c$ is the total kinetic energy, $E_p$ is the total potential energy, $q \in \mathbb{R}$ is the vector of joint positions and $\dot{q} \in \mathbb{R}$ is the vector of joint velocities.

The geometric coordinates of link 1 are given by the following equation:

$$\begin{aligned} x_1 &= l_{c1}\sin(q_1) \\ y_1 &= -l_{c1}\cos(q_1) \end{aligned} \tag{7}$$

The velocity vector $v_1$ of the same link is given by:

$$v_1 = \begin{pmatrix} \dot{x}_1 \\ \dot{y}_1 \end{pmatrix} = \begin{pmatrix} l_{c1}\cos(q_1)\dot{q}_1 \\ l_{c1}\sin(q_1)\dot{q}_1 \end{pmatrix} \tag{8}$$

The corresponding kinetic energy for the first link is given by the formula (9):

$$E_{c1}(q,\dot{q}) = \frac{1}{2} m_1 v_1^T v_1 + \frac{1}{2} I_1 \dot{q}_1^2 = \frac{m_1}{2} l_{c1}^2 \dot{q}_1^2 + \frac{1}{2} l_{c2}^2 \dot{q}_1^2 \tag{9}$$

On the other hand, the geometric parameters of link 2, expressed in the $(x-y)$ plane, are given by the following equations:

$$\begin{aligned} x_2 &= l_1 \sin(q_1) + l_{c2} \sin(q_1 + q_2) \\ y_2 &= -l_1 \cos(q_1) - l_{c2} \cos(q_1 + q_2) \end{aligned} \tag{10}$$

The velocity vector $v_2$ of the same link is given by:

$$v_2 = \begin{pmatrix} \dot{x}_2 \\ \dot{y}_2 \end{pmatrix} = \begin{pmatrix} l_1 \cos(q_1)\dot{q}_1 + l_{c2} \cos(q_1 + q_2)\left[\dot{q}_1 + \dot{q}_2\right] \\ l_1 \sin(q_1)\dot{q}_1 + l_{c2} \sin(q_1 + q_2)\left[\dot{q}_1 + \dot{q}_2\right] \end{pmatrix} \tag{11}$$

The equation for the kinetic energy of joint 2 is as follows:

$$\begin{aligned} E_{c2}(q,\dot{q}) &= \frac{1}{2} m_2 v_2^T v_2 + \frac{1}{2} I_1 \left[\dot{q}_1 + \dot{q}_2\right]^2 \\ &= \frac{m_2}{2} l_1^2 \dot{q}_1^2 + \frac{m_2}{2} l_{c2}^2 \left[\dot{q}_1^2 + 2\dot{q}_1\dot{q}_2 + \dot{q}_2^2\right] + m_2 l_1 l_{c2} \left[\dot{q}_1^2 + \dot{q}_1\dot{q}_2\right] \cos(q_2) + \frac{1}{2} I_2 \left[\dot{q}_1 + \dot{q}_2\right]^2 \end{aligned} \tag{12}$$

The total kinetic energy is described by the equation (13):

$$\begin{aligned} E_c(q,\dot{q}) &= E_{c1}(q,\dot{q}) + E_{c2}(q,\dot{q}) \\ &= \frac{1}{2}(m_1 l_{c1}^2 + m_2 l_1^2)\dot{q}_1^2 + \frac{1}{2} m_2 l_{c2}^2 \left[\dot{q}_1^2 + 2\dot{q}_1\dot{q}_2 + \dot{q}_2^2\right] + m_2 l_1 l_{c2} \left[\dot{q}_1^2 + \dot{q}_1\dot{q}_2\right] \cos(q_2) \\ &+ \frac{1}{2} I_1 \dot{q}_1^2 + \frac{1}{2} I_2 \left[\dot{q}_1 + \dot{q}_2\right]^2 \end{aligned} \tag{13}$$

Similarly, the total potential energy $E_p$ is expressed by the formula (14):

$$E_p(q) = E_{p1}(q) + E_{p2}(q) \tag{14}$$

where $E_{p1}(q)$ and $E_{p2}(q)$ represent the potential energies associated with masses $m_1$ and $m_2$ respectively, they are defined by the following two equations:

$$E_{p1}(q) = m_1 \times g \times h_1 = -m_1 \times g \times l_{c1} \cos(q_1) \tag{15}$$

$$\begin{aligned} E_{p2}(q) &= m_2 \times g \times h_2 = m_2 \times g \times \left[-l_1 \cos(q_1) - l_{c2} \cos(q_1 + q_2)\right] \\ &= -m_2 \times g \times l_1 \cos(q_1) - m_2 \times g \times l_{c2} \cos(q_1 + q_2) \end{aligned} \tag{16}$$

The Lagrangian, for the dynamic modeling of the robot, is described as follows:

$$\begin{aligned} L(q,\dot{q}) &= E_c(q,\dot{q}) - E_p(q) \\ &= \frac{1}{2}\left(m_1 l_{c1}^2 + m_2 l_1^2\right)\dot{q}_1^2 + \frac{1}{2} m_2 l_{c2}^2 \left(\dot{q}_1^2 + 2\dot{q}_1\dot{q}_2 + \dot{q}_2^2\right) + m_2 l_1 l_{c2} \cos(q_2)\left[\dot{q}_1^2 + \dot{q}_1\dot{q}_2\right] \\ &+ \left(m_1 l_{c1} + m_2 l_1\right) g \cos(q_1) + m_2 g l_{c2} \cos(q_1 + q_2) + \frac{1}{2} I_1 \dot{q}_1^2 + \frac{1}{2} I_2 \left[\dot{q}_1 + \dot{q}_2\right]^2 \end{aligned} \tag{17}$$

From this final equation, we deduce the following expressions:

$$\frac{\partial L}{\partial \dot{q}_1} = \left[m_1 l_{c1}^2 + m_2 l_1^2 + m_2 l_{c2}^2 + 2 m_2 l_1 l_{c2} \cos(q_2)\right]\dot{q}_1 + \left[m_2 l_{c2}^2 + m_2 l_1 l_{c2} \cos(q_2)\right]\dot{q}_2 + I_1 \dot{q}_1 + I_2 \left[\dot{q}_1 + \dot{q}_2\right]$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_1}\right)=\left[m_1 l_{c1}^2+m_2 l_1^2+m_2 l_{c2}^2+2m_2 l_1 l_{c2}\cos(q_2)\right]\ddot{q}_1+\left[m_2 l_{c2}^2+m_2 l_1 l_{c2}\cos(q_2)\right]\ddot{q}_2$$
$$-2m_2 l_1 l_{c2}\sin(q_2)\dot{q}_1\dot{q}_2-m_2 l_1 l_{c2}\sin(q_2)\dot{q}_2^2+I_1\ddot{q}_1+I_2\left[\ddot{q}_1+\ddot{q}_2\right]$$

$$\frac{\partial L}{\partial q_1}=-(m_1 l_{c1}+m_2 l_1)g\sin(q_1)-m_2 g l_{c2}\sin(q_1+q_2)$$
$$\frac{\partial L}{\partial \dot{q}_2}=m_2 l_{c2}^2\dot{q}_1+m_2 l_{c2}^2\dot{q}_2+m_2 l_1 l_{c2}\cos(q_2)\dot{q}_1+I_2\left(\dot{q}_1+\dot{q}_2\right)$$
$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_2}\right)=m_2 l_{c2}^2\ddot{q}_1+m_2 l_{c2}^2\ddot{q}_2+m_2 l_1 l_{c2}\cos(q_2)\ddot{q}_1-m_2 l_1 l_{c2}\sin(q_2)\dot{q}_1\dot{q}_2+I_2\left(\ddot{q}_1+\ddot{q}_2\right)$$
$$\frac{\partial L}{\partial q_2}=-m_2 l_1 l_{c2}\sin(q_2)\left[\dot{q}_1^2+\dot{q}_1\dot{q}_2\right]-m_2 g l_{c2}\sin(q_1+q_2)$$

The dynamic model is:

$$\tau_1=\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_1}\right)-\frac{\partial L}{\partial q_1}\ \text{ and }\ \tau_2=\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_2}\right)-\frac{\partial L}{\partial q_2}$$

$$\tau_1=\left[m_1 l_{c1}^2+m_2 l_1^2+m_2 l_{c2}^2+2m_2 l_1 l_{c2}\cos(q_2)+I_1+I_2\right]\ddot{q}_1+\left[m_2 l_{c2}^2+m_2 l_1 l_{c2}\cos(q_2)+I_2\right]\ddot{q}_2$$
$$-2m_2 l_1 l_{c2}\sin(q_2)\dot{q}_1\dot{q}_2-m_2 l_1 l_{c2}\sin(q_2)\dot{q}_2^2+\left[m_1 l_{c1}+m_2 l_1\right]g\sin(q_1)+m_2 g l_{c2}\sin(q_1+q_2)$$

$$\tau_2=\left[m_2 l_{c2}^2+m_2 l_1 l_{c2}\cos(q_2)+I_2\right]\ddot{q}_1+\left[m_2 l_{c2}^2+I_2\right]\ddot{q}_2+m_2 l_1 l_{c2}\sin(q_2)\dot{q}_2^2+m_2 g l_{c2}\sin(q_1+q_2)$$

Given that $\tau_1$ and $\tau_2$ are the torques provided to the actuators of joints 1 and 2, respectively, the dynamic model is expressed in matrix form as equation (18):

$$\underbrace{\begin{pmatrix} M_{11}(q) & M_{12}(q) \\ M_{21}(q) & M_{22}(q)\end{pmatrix}}_{M(q)}\ddot{q}+\underbrace{\begin{pmatrix} C_{11}(q,\dot{q}) & C_{12}(q,\dot{q}) \\ C_{21}(q,\dot{q}) & C_{22}(q,\dot{q})\end{pmatrix}}_{C(q,\dot{q})}\dot{q}+\underbrace{\begin{pmatrix} g_1(q) \\ g_2(q)\end{pmatrix}}_{g(q)}=\tau \qquad (18)$$

$$M_{11}(q)=m_1 l_{c1}^2+m_2\left[l_1^2+l_{c2}^2+2l_1 l_{c2}\cos(q_2)\right]+I_1+I_2\quad,\quad M_{12}(q)=m_2 l_{c2}^2+m_2 l_1 l_{c2}\cos(q_2)+I_2$$
$$M_{21}(q)=m_2 l_{c2}^2+m_2 l_1 l_{c2}\cos(q_2)+I_2,\ M_{22}(q)=m_2 l_{c2}^2+I_2$$

$C_{11}(q,\dot{q})=-m_2 l_1 l_{c2}\sin(q_2)\dot{q}_2$ , $C_{12}(q,\dot{q})=-m_2 l_1 l_{c2}\sin(q_2)\left[\dot{q}_1+\dot{q}_2\right]$ , $C_{21}(q,\dot{q})=m_2 l_1 l_{c2}\sin(q_2)\dot{q}_1$ and $C_{22}(q,\dot{q})=0$

$g_1(q)=\left(m_1 l_{c1}+m_2 l_1\right)g\sin(q_1)+m_2 g l_{c2}\sin(q_1+q_2)$ and $g_2(q)=m_2 g l_{c2}\sin(q_1+q_2)$

## 4. Simulation results

To validate the robustness of this PD type ILC approach, two reference trajectories were selected: a sinusoidal trajectory and a complex exponential sinusoidal trajectory. The simulations for these two reference trajectories were conducted using the following parameters of the PD controller:

$k_p=\begin{pmatrix}200 & 0\\ 0 & 200\end{pmatrix}$, $k_d=\begin{pmatrix}300 & 0\\ 0 & 300\end{pmatrix}$, simulation time t = 10s, number of iterations k=20

**1st scenario: Sinusoidal trajectory**

The desired trajectories $q_{1d}$ and $q_{2d}$ are described by the following equation:

$$q_{1d} = \sin(3t) \text{ and } q_{2d} = \cos(3t) \tag{19}$$

Figure 3 shows the temporal evolution of the desired trajectories:

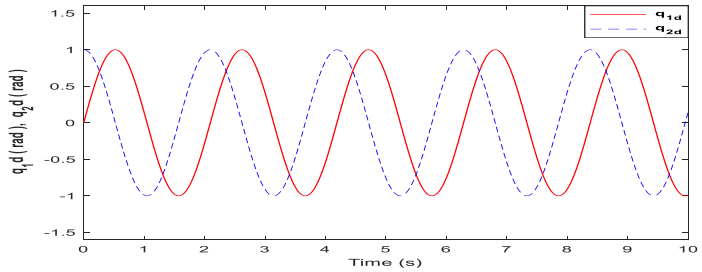


**Figure 3.** Desired positions to be followed by both joints (Own research)

The desired trajectories of velocities $\dot{q}_1$ and $\dot{q}_2$ are obtained by deriving the outputs described by equation (19). These trajectories are given by equation (20):

$$\dot{q}_{1d} = 3\cos(3t) \text{ and } \dot{q}_{2d} = -3\sin(3t) \tag{20}$$

Figure 4 illustrates the temporal evolution of the desired velocities $\dot{q}_{1d}$ and $\dot{q}_{2d}$.

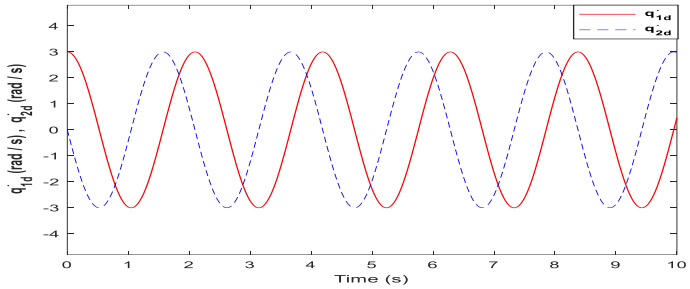


**Figure 4.** Desired velocities to be followed by both joints (Own research)

The position errors of joints 1 and 2 are respectively represented by $e_1(t) = q_{1d}(t) - q_1(t)$ and $e_2(t) = q_{2d}(t) - q_2(t)$. The corresponding velocity errors for these joints are given by $e_3(t) = \dot{q}_{1d}(t) - \dot{q}_1(t)$ and $e_4(t) = \dot{q}_{2d}(t) - \dot{q}_2(t)$.

Figures 5 and 6 show that the robot controlled by the PD type ILC follows the desired trajectories perfectly. This observation confirms the effectiveness and robustness of the proposed control.

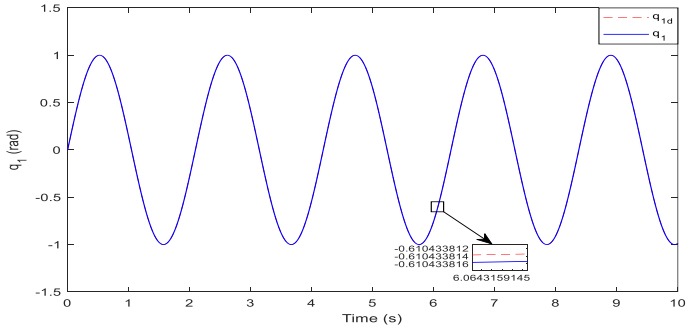


**Figure 5.** Position tracking for the first joint $q_1$ (Own research)

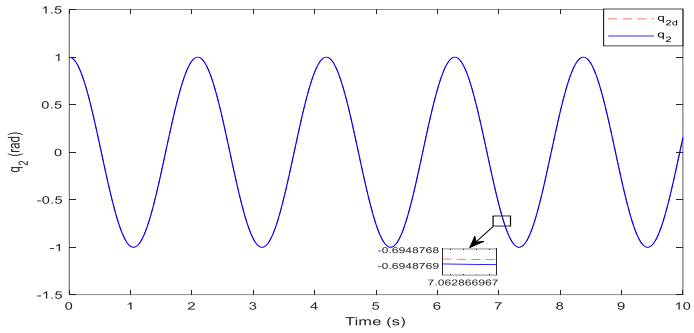


**Figure 6.** Position tracking for the second joint $q_2$ (Own research)

Figures 7 and 8 depict the temporal evolution of the velocities $\dot{q}_1$ and $\dot{q}_2$, shown in solid blue lines, with their references indicated by dashed red lines. The robustness of PD-type ILC in trajectory tracking is demonstrated by the robot's actual velocities that perfectly converge towards their references.

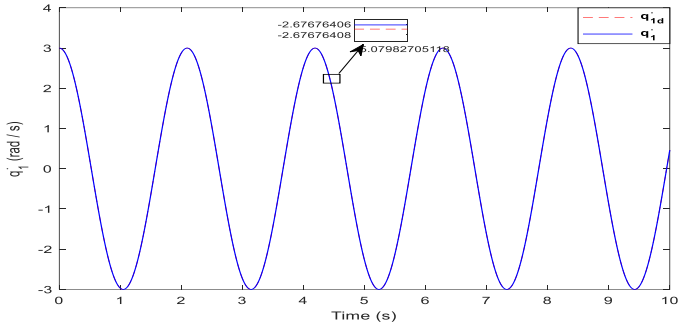


**Figure 7.** Velocity tracking for the first joint $\dot{q}_1$ (Own research)

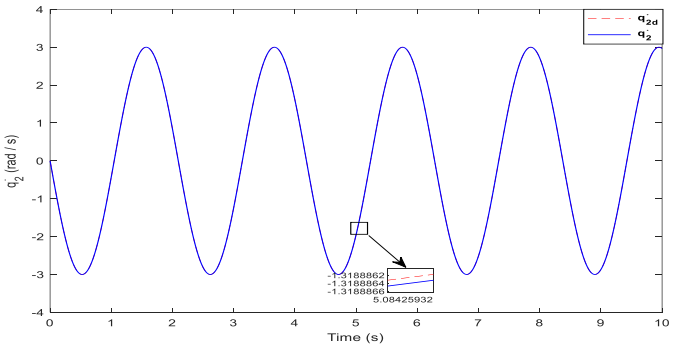


**Figure 8.** Velocity tracking for the second joint $\dot{q}_2$ (Own research)

Figure 9 depicts the temporal evolution of trajectory tracking errors in terms of position. The position error $e_1$ of joint $q_1$ converges to $10^{-9}$. This error practically converges to zero. The position error $e_2$ of joint $q_2$ oscillates between $8.10^{-8}$ and $-5.10^{-8}$, with a tendency to converge towards zero. These results demonstrate the high precision of the robot in its movements in $q_1$ and $q_2$.

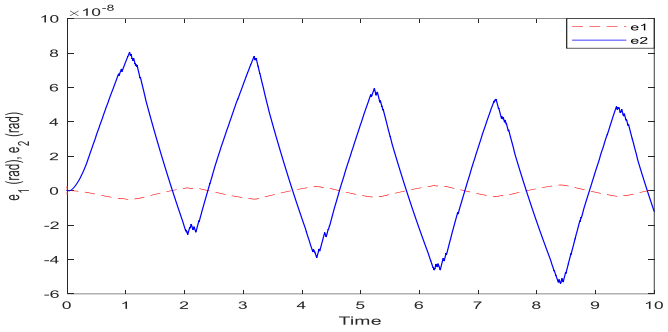


**Figure 9.** Position errors for both joints ($e_1$ and $e_2$) (Own research)

Figure 10 illustrates that the velocity errors ($e_3$ and $e_4$) for $\dot{q}_1$ and $\dot{q}_2$ are very small. Their numerical values are on the order of $10^{-5}$; we consider that these errors converge towards zero. The Pelican robot follows these reference trajectories with high precision.

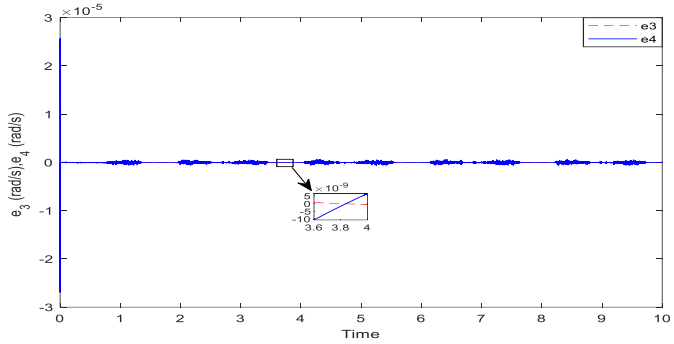


**Figure 10.** Velocity errors for both joints ($e_3$ and $e_4$) (Own research)

**2nd Scenario: Exponential-Sinusoidal Trajectory**

In this section, we take a closer look at how robust the PD-type Iterative Learning Control is when it comes to tracking complex trajectories. In particular, we examine how well the system holds up when dealing with disturbances and uncertainties in its environment. We also assess how the PD-type ILC improves tracking accuracy over multiple iterations. This analysis is important to ensure that the robot can handle sophisticated tasks with reliability and precision, even under changing or unexpected conditions. To test this, we use an exponential-sinusoidal reference trajectory for position tracking. It is defined by the following equation:

$$q_{1d} = b_1\left[1-e^{-2.0t^2}\right]+c_1\left[1-e^{-2.0t^2}\right]\sin\omega_1 t\,,\ q_{2d} = b_2\left[1-e^{-2.0t^2}\right]+c_2\left[1-e^{-2.0t^2}\right]\sin\omega_2 t \quad (21)$$

where $b_1 = \pi/4$ [rad], $c_1 = \pi/9$ [rad] and $\omega_1 = 4$ [rad/s] are the parameters for the desired position reference of the first link, and $b_2 = \pi/3$ [rad], $c_2 = \pi/6$ [rad] and $\omega_2 = 3$ [rad/s] correspond to the parameters for the desired position reference of the second link.

Figure 11 shows the profiles of the desired position trajectories $q_{1d}$ and $q_{2d}$.

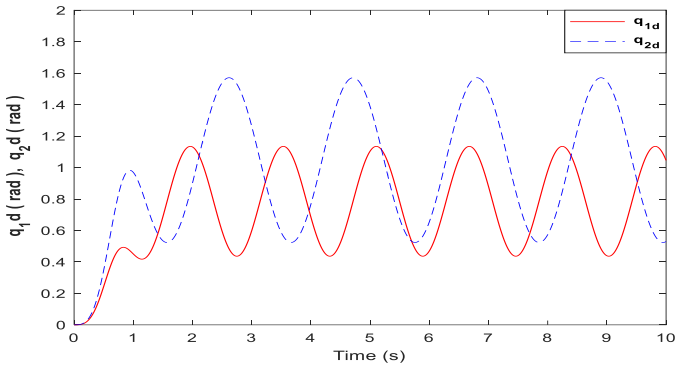


**Figure 11.** Desired positions to be tracked by the two joints (Own research)

By utilizing the expressions for the desired position trajectories provided in equation (21), we can derive the analytical formulas for the reference velocity trajectories, as determined by the following equation:

$$\begin{aligned}\dot{q}_{1d} &= 6b_1t^2e^{-2t^3} + 6c_1t^2e^{-2t^3}\sin(\omega_1 t) + \left[c_1 - c_1e^{-2t^3}\right]\cos(\omega_1 t)\omega_1 \\ \dot{q}_{2d} &= 6b_2t^2e^{-2t^3} + 6c_2t^2e^{-2t^3}\sin(\omega_2 t) + \left[c_2 - c_2e^{-2t^3}\right]\cos(\omega_2 t)\omega_2\end{aligned} \tag{22}$$

Figure 12 shows the temporal evolution of the desired velocities $\dot{q}_{1d}$ and $\dot{q}_{2d}$.

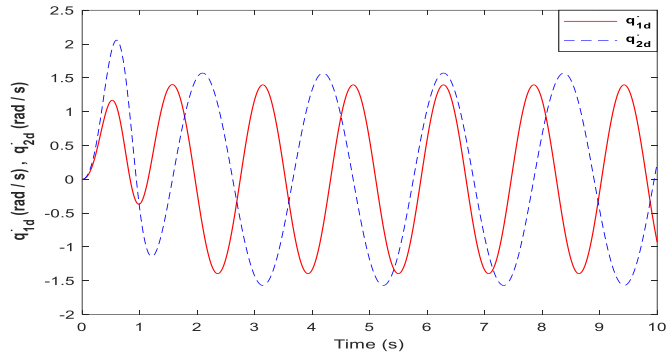


**Figure 12.** Desired velocities to be followed by both joints (Own research)

Figure 13 shows the temporal evolution of the joint $q_1$ relative to its reference, represented by a red dashed line. We observe that $q_1$ closely follows its reference, highlighting the effectiveness of the PD-type ILC.

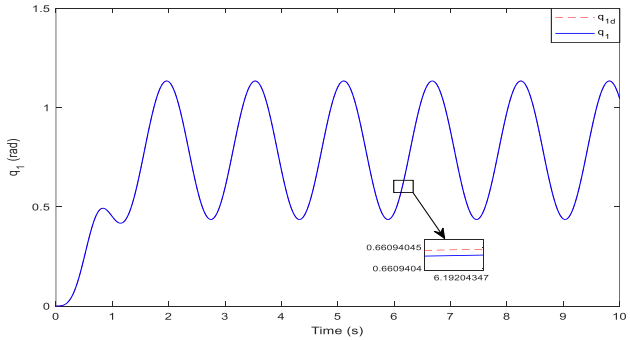


**Figure 13.** Position tracking for the first joint $q_1$ (Own research)

Figure 14 presents the temporal evolution of the joint $q_2$ relative to its reference $q_{2d}$, depicted by a red dashed line. After 6.44 seconds, it is clear that $q_2$ tracks its reference. The

complexity of the trajectory causes the transient response to be slow, showing the robustness of the PD-type ILC method used with the Pelican robot.

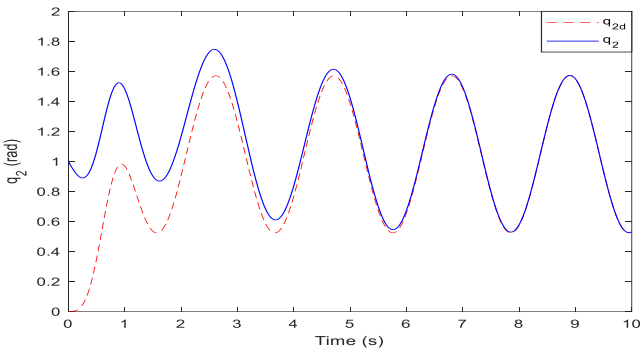


**Figure 14.** Position tracking for the second joint $q_2$ (Own research)

Figure 15 shows excellent velocity trajectory tracking with very high precision with reference. This demonstrates how well the Pelican robot's control system works.

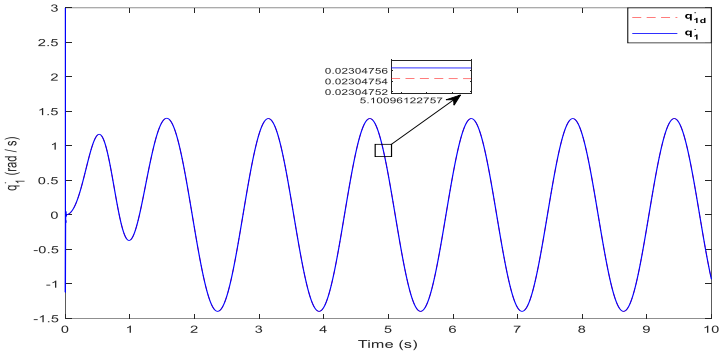


**Figure 15.** Velocity tracking for the first joint $\dot{q}_1$ (Own research)

In Figure 16, the output $\dot{q}_2$ is depicted alongside its reference $\dot{q}_{2d}$. A slight phase lag between $\dot{q}_2$ and $\dot{q}_{2d}$ is observed after 4.5 seconds, with a very low but acceptable steady-state error. From this point onward, $\dot{q}_2$ closely tracks its reference with high precision, demonstrating the Pelican robot's ability to achieve virtually zero steady-state velocity error.

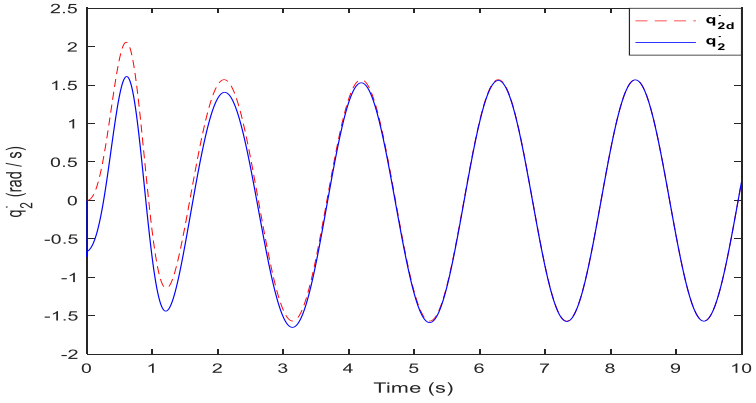


**Figure 16.** Velocity tracking for the second joint $\dot{q}_2$ (Own research)

In Figure 17, the trajectory tracking errors in terms of position ($e_1$ and $e_2$) are depicted. The position error $e_1$ of joint $q_1$ quickly converges to zero, highlighting the precision of the Pelican robot in trajectory tracking. As for the position error $e_2$ of joint $q_2$, it only converges to zero after

6.44 seconds. This observation is due to the fact that, as shown in figure 14, the output follows its reference perfectly only after 6.44 seconds, with zero steady-state error.

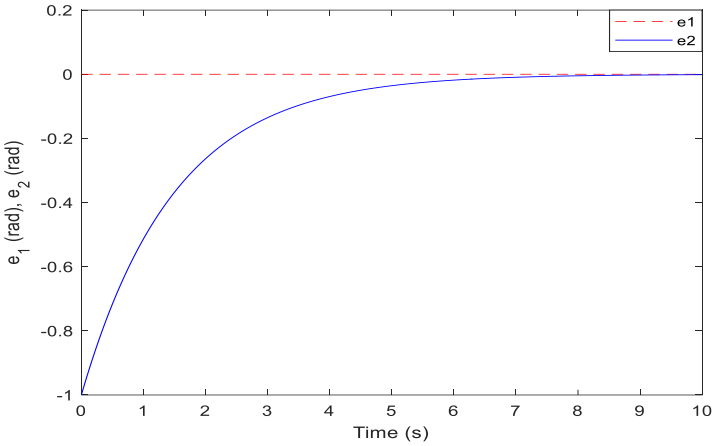


**Figure 17.** Position errors for both joints ($e_1$ and $e_2$) (Own research)

Figure 18 displays the trajectory tracking errors in terms of velocity ($e_3$ and $e_4$) for both joints $q_1$ and $q_2$. The error $e_3$ converges to zero, while the velocity error $e_4$, representing the error for joint $q_2$, converges to zero only after 5.5 seconds.

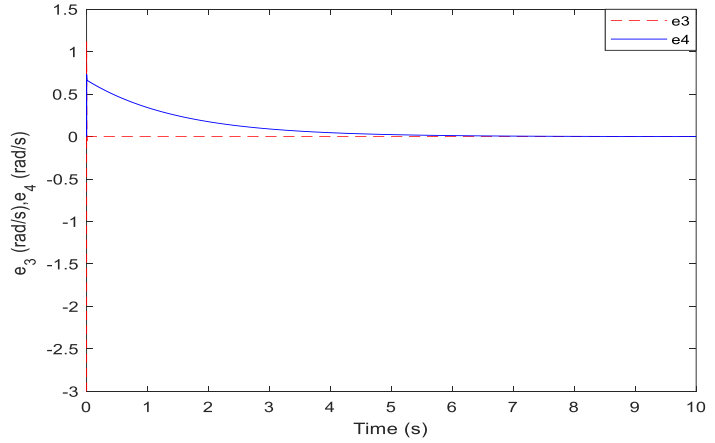


**Figure 18.** Velocity errors for both joints ($e_3$ and $e_4$) (Own research)

## 5. Conclusions

In conclusion, the trajectory tracking problem for the Pelican robot has been effectively resolved through the application of the Iterative Learning Control combined with the proportional-derivative control (PD-type ILC). The findings show that the system tracks both joint positions and velocities with impressive accuracy. For joints 1 and 2, the errors were very small, reflecting near-perfect alignment with the desired trajectories. This performance is mainly due to Proportional-Derivative (PD)-type Iterative Learning Control (ILC), which adjusts the control input over time by learning from previous attempts, gradually reducing errors and improving precision. Further, the controller's proportional-derivative component allows the system to stay stable and fast by offering prompt corrections when deviations arise.These findings demonstrate the resilience and dependability of the PD-type ILC, even in intricate or dynamic settings. It enables the Pelican robot to adapt in real time to uncertainties or disturbances without sacrificing its trajectory tracking accuracy.

This achievement suggests that the technique could be used for other robots that require steady, accurate control. To summarize, the study demonstrates that the PD-type ILC is a robust and dependable method for resolving intricate trajectory tracking issues and has a lot of potential for application in numerous robotic systems in the future.

# REFERENCES

Amieur, T., Taibi, D., Bechouat, M., Kahla, S. & Sedraoui, M. (2022) Fuzzy Sliding Mode with adaptive gain control for nonlinear MIMO systems. *Romanian Journal of Information Technology and Automatic Control*. 32(3), 95–108. doi: 10.33436/v32i3y202208

Bien, Z. & Xu, J.-X. (1998) *Iterative Learning Control: Analysis, Design, Integration and Applications*. Springer.doi:10.1007/978-1-4615-5629-9.

Bouakrif, F. & Zasadzinski, M. (2016) Trajectory tracking control for perturbed robot manipulators using iterative learning method. *The International Journal of Advanced Manufacturing Technology*. 87(1), 2013–2022. doi:10.1007/s00170-016-8550-3.

Bouakrif, F., Boukhetala, D. & Boudjema, F. (2007) Iterative learning control for robot manipulators. *Archives of Control Sciences*. 17(1), 57–69.

Bristow, D. A., Tharayil, M. & Alleyne, A. G. (2006) A survey of iterative learning control. *IEEE Control Systems Magazine*. 26, 96–114. doi:10.1109/MCS.2006.1636313.

Chong, S.-H., Chan, C.-Y., Sato, K., Sakthivelu, V. & Loh, S.-L. (2017) Modified-PID Control with Feedforward Improvement for 1-Degree-of-Freedom Pneumatic Muscle Actuated System. *Jurnal Teknologi (Sciences & Engineering)*. 79(5-2), 51–57. doi:10.11113/jt.v79.11283.

Cojuhari, I. (2011) Tuning Controllers to the Identical Model Objects in the Cascade Control System. *Romanian Journal of Information Technology and Automatic Control*. 21(1), 71-78.

Dehghani, A. & Khodadadi, H. (2016) Self-Tuning PID controller design using fuzzy logic for a single-link flexible joint robot manipulator. *Jurnal Teknologi (Sciences & Engineering)*. 78(6–13), 115–120. doi:10.11113/jt.v78.9282.

Kelly, R., Santibáñez Dávila, V. & Loría, A. (2005) *Control of Robot Manipulators in Joint Space*. Springer. doi:10.1007/b135572.

Liu, Y., Wu, Z., Lai, J., Li, J. & Xie, S. (2022) Lyapunov-Based Iterative Learning Feedback Control Design for Parabolic MIMO PDEs with Time-Varying Delays. *IET Control Theory & Applications*. 16(2), 799–815. doi:10.1049/cth2.12272.

Meng, T. & He, W. (2020) *Iterative Learning Control for Flexible Structures.* Springer. doi:10.1007/978-981-15-2784-5.

Mohamad Yuden, M. A., Md Ghazaly, M., Che Amran, A., Jamaludin, I. W., Hui Yee, K., Yaacob, M. R., Abdullah, Z. & Chin Kiat, Y. (2017) Positioning Control Performances of a Robotic Hand System. *Jurnal Teknologi (Sciences & Engineering).* 79(1), 1–23. doi:10.11113/jt.v79.8726

Norouzi, A. & Koch, C.R. (2020) Integration of PD-type Iterative Learning Control with Adaptive Sliding Mode Control. *IFAC-PapersOnLine*. 53, 6213–6218. doi:10.1016/j.ifacol.2020.12.1717.

Ouyang, P.R. & Pipatpaibul, P. (2010) Iterative Learning Control: A Comparison Study. Proceedings of the *ASME 2010 International Mechanical Engineering Congress and Exposition*. *Volume 8: Dynamic Systems and Control, Parts A and B. Vancouver, British Columbia, Canada. November 12–18, 2010*. pp. 939-945. doi:10.1115/IMECE2010-37161.

Riaz, S., Qi, R., Tutsoy, O. & Iqbal, J. (2023) A novel adaptive PD-type iterative learning control of the PMSM servo system with the friction uncertainty in low speeds. *PLOS ONE*. 18(1), e0279253, 1-22. doi:10.1371/journal.pone.0279253.

Saidi, I. & Touati, N. (2023) Learning control for trajectory tracking of nonlinear inertia wheel inverted pendulum. *Revue Roumaine des Sciences Techniques. Série Électrotechnique et Énergétique*. 68(4), 423–429. doi:10.59277/RRST-EE.2023.4.17

Sciavicco, L. & Siciliano, B. (2000) Modelling and Control of Robot Manipulators. *Measurement Science and Technology*. 11(12), 131. doi:10.1088/0957-0233/11/12/709.

Shen, D. & Wang, Y. (2014) Survey on stochastic iterative learning control. *Journal of Process Control*. 24(12), 64–77. doi:10.1016/j.jprocont.2014.04.013.

Uchiyama, M. (1978) Formation of High-Speed Motion Pattern of a Mechanical Arm By Trial. *Transactions of the Society of Instrument and Control Engineers*. 14(6), 706–712. doi:10.9746/SICETR1965.14.706.

Umar, S.N.H. & Bakar, E.A. (2014) Study on Trajectory Motion and Computational Analysis of Robot Manipulator. *Jurnal Teknologi (Sciences & Engineering).* 67(1), 53–59. doi:10.11113/jt.v67.2206.

Wang, L., Dong, L., Chen, Y., Wang, K. & Gao, F. (2022) Iterative Learning Control for Actuator Fault Uncertain Systems. *Symmetry*. 14(10), 1969 1–17. doi:10.3390/sym14101969.

Wei, B. (2018) Adaptive Control Design and Stability Analysis of Robotic Manipulators. *Actuators*. 7(4), 1-17. doi:10.3390/act7040089.

Xu, J.-X. & Qu, Z. (1998) Robust Iterative Learning Control for a Class of Nonlinear Systems. *Automatica*. 34(8), 983–988. doi:10.1016/S0005-1098(98)00036-3.

Xu, S., Zhang, C. & Mohammadzadeh, A. (2023) Type-3 fuzzy control of robotic manipulators. *Symmetry*. 15(2), 483, 1–18. doi:10.3390/sym15020483.

Yu, Q., Hou, Z. & Xu, J.-X. (2018) D-Type ILC based dynamic modeling and norm optimal ILC for high-speed trains. *IEEE Transactions on Control Systems Technology*. 26, 652–663. doi:10.1109/TCST.2017.2692730.

***

**Khouloud GUESMI** has an applied bachelor's degree in Industrial Systems Electronics from the Higher Institute of Applied Sciences and Technology of Mateur (ISSAT Mateur). Then, she obtained a professional master's degree in Industrial Systems Control from (ISSAT Mateur) and in 2023 she got a research master's degree in Signal, Image, and Advanced Automation from the National Higher School of Engineers of Tunis (ENSIT). She is currently a doctoral student at Tunisia's National School of Engineers of Tunis (ENSIT). Robot control is the main topic of her research.

***

**Imen SAIDI** is a Professor of Electrical Engineering at the National Higher School of Engineers of Tunis. She graduated from the National School of Engineers of Tunis (ENIT) with a degree in Electrical Engineering in 2006, a Master's degree in Automatic Control in 2007, and a Ph.D. and University Habilitation (HDR) in Electrical Engineering in 2011 and 2019, respectively. She is also a researcher at the Automatic Control Research Laboratory (L.A.R.A) at ENIT. Her research is mostly about linear and nonlinear control, which has several uses in robotics, such as underactuated robotics, overactuated robotics, parallel robotics, and exoskeletons. She also looks at and puts together complicated systems utilizing both traditional and non-traditional methods (square and non-square systems), designs for intelligent control, and renewable energy.