Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2010, Article ID 639801, 34 pages
doi:10.1155/2010/639801

Review Article

Some Applications of Fractional Calculus in Engineering

**J. A. Tenreiro Machado, Manuel F. Silva,
Ramiro S. Barbosa, Isabel S. Jesus, Cecília M. Reis,
Maria G. Marcos, and Alexandra F. Galhano**

Institute of Engineering of Porto, Porto, Portugal

Correspondence should be addressed to J. A. Tenreiro Machado, jtm@isep.ipp.pt

Received 9 June 2009; Accepted 29 July 2009

Academic Editor: Jose Balthazar

Copyright © 2010 J. A. Tenreiro Machado et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Fractional Calculus (FC) goes back to the beginning of the theory of differential calculus. Nevertheless, the application of FC just emerged in the last two decades, due to the progress in the area of chaos that revealed subtle relationships with the FC concepts. In the field of dynamical systems theory some work has been carried out but the proposed models and algorithms are still in a preliminary stage of establishment. Having these ideas in mind, the paper discusses FC in the study of system dynamics and control. In this perspective, this paper investigates the use of FC in the fields of controller tuning, legged robots, redundant robots, heat diffusion, and digital circuit synthesis.

1. Introduction

The generalization of the concept of derivative $D^{\alpha}[f(x)]$ to noninteger values of α goes back to the beginning of the theory of differential calculus. In fact, Leibniz, in his correspondence with Bernoulli, L'Hôpital and Wallis (1695), had several notes about the calculation of $D^{1/2}[f(x)]$. Nevertheless, the development of the theory of Fractional Calculus (FC) is due to the contributions of many mathematicians such as Euler, Liouville, Riemann, and Letnikov [1–3].

The FC deals with derivatives and integrals to an arbitrary order (real or, even, complex order). The mathematical definition of a derivative/integral of fractional order has been the subject of several different approaches [1–3]. For example, the Laplace definition of a fractional derivative of a signal $x(t)$ is

$$D^{\alpha}x(t) = L^{-1}\left\{ s^{\alpha}X(s) - \sum_{k=0}^{n-1} s^{k} D^{\alpha-k-1}x(t)|_{t=0} \right\}, \tag{1.1}$$

where $n-1<\alpha\leq n$, $\alpha>0$. The Grünwald-Letnikov definition is given by ($\alpha\in\Re$):

$$D^{\alpha}x(t)=\lim_{h\to 0}\left[\frac{1}{h^{\alpha}}\sum_{k=0}^{\infty}(-1)^{k}\binom{\alpha}{k}x(t-kh)\right],$$
$$\binom{\alpha}{k}=\frac{\Gamma(\alpha+1)}{\Gamma(k+1)\Gamma(\alpha-k+1)},\tag{1.2}$$

where Γ is the Gamma function and h is the time increment. However, (1.2) shows that fractional-order operators are "global" operators having a memory of all past events, making them adequate for modeling memory effects in most materials and systems.

The Riemann-Liouville definition of the fractional-order derivative is ($\alpha>0$):

$${}_{a}D_{t}^{\alpha}f(t)=\frac{1}{\Gamma(n-\alpha)}\frac{d^{n}}{dt^{n}}\int_{a}^{t}\frac{f(\tau)}{(t-\tau)^{\alpha-n+1}}d\tau,\quad n-1<\alpha<n,\tag{1.3}$$

where $\Gamma(x)$ is the Gamma function of x.

Based on the proposed definitions it is possible to calculate the fractional-order integrals/derivatives of several functions (Table 1). Nevertheless, the problem of devising and implementing fractional-order algorithms is not trivial and will be the matter of the following sections.

In recent years FC has been a fruitful field of research in science and engineering [1–6]. In fact, many scientific areas are currently paying attention to the FC concepts and we can refer its adoption in viscoelasticity and damping, diffusion and wave propagation, electromagnetism, chaos and fractals, heat transfer, biology, electronics, signal processing, robotics, system identification, traffic systems, genetic algorithms, percolation, modeling and identification, telecommunications, chemistry, irreversibility, physics, control systems as well as economy, and finance [7–18].

Bearing these ideas in mind, Sections 2–6 present several applications of FC in science and engineering. In Section 2, it is presented the application of FC concepts to the tuning of PID controllers and, in Section 3, the application of a fractional-order PD controller in the control of the leg joints of a hexapod robot. Then in Section 4, it is presented the fractional dynamics in the trajectory control of redundant manipulators. Next, in Section 5, it is introduced the fractional characteristics of heat diffusion along a media and, in Section 6 it is shown the application of FC to circuit synthesis using evolutionary algorithms. Finally, the main conclusions are presented in Section 7.

2. Tuning of PID Controllers Using Fractional Calculus Concepts

The PID controllers are the most commonly used control algorithms in industry. Among the various existent schemes for tuning PID controllers, the Ziegler-Nichols (Z-N) method is the most popular and is still extensively used for the determination of the PID parameters. It is well known that the compensated systems, with controllers tuned by this method, have generally a step response with a high percent overshoot. Moreover, the Z-N heuristics are only suitable for plants with monotonic step response.

Table 1: Fractional-order integrals of several functions.

$\varphi(x),\ x \in \Re$	$(I_+^{\alpha}\varphi)(x),\ x \in \Re,\ \alpha \in C$
$(x-a)^{\beta-1}$	$\dfrac{\Gamma(\beta)}{\Gamma(\alpha+\beta)}(x-a)^{\alpha+\beta-1},\ \mathrm{Re}(\beta) > 0$
$e^{\lambda x}$	$\lambda^{-\alpha}e^{\lambda x},\ \mathrm{Re}(\lambda) > 0$
$\begin{cases}\sin(\lambda x)\\ \cos(\lambda x)\end{cases}$	$\lambda^{-\alpha}\begin{cases}\sin(\lambda x - \alpha\pi/2),\\ \cos(\lambda x - \alpha\pi/2),\end{cases}\ \lambda > 0,\ \mathrm{Re}(\alpha) > 1$
$e^{\lambda x}\begin{cases}\sin(\gamma x)\\ \cos(\gamma x)\end{cases}$	$\dfrac{e^{\lambda x}}{(\lambda^2+\gamma^2)^{\alpha/2}}\begin{cases}\sin(\gamma x - \alpha\phi), & \phi = \arctan(\gamma/\lambda)\\ \cos(\gamma x - \alpha\phi), & \gamma > 0,\ \mathrm{Re}(\lambda) > 1\end{cases}$

In this section, we study a methodology for tuning PID controllers such that the response of the compensated system has an almost constant overshoot defined by a prescribed value. The proposed method is based on the minimization of the integral of square error (ISE) between the step responses of a unit feedback control system, whose open-loop transfer function $L(s)$ is given by a fractional-order integrator and that of the PID compensated system [7].

Figure 1 illustrates the fractional-order control system that will be used as reference model for the tuning of PID controllers. The open-loop transfer function $L(s)$ is defined as ($\alpha \in \Re^+$):

$$L(s) = \left(\frac{\omega_c}{s}\right)^{\alpha}, \tag{2.1}$$

where ω_c is the gain crossover frequency, that is, $|L(j\omega_c)| = 1$. The parameter α is the slope of the magnitude curve, on a log-log scale, and may assume integer as well as noninteger values. In this study we consider $1 < \alpha < 2$, such that the output response may have a fractional oscillation (similar to an underdamped second-order system). This transfer function is also known as the Bode's ideal loop transfer function since Bode studies on the design of feedback amplifiers in the 1940s [19].

The Bode diagrams of amplitude and phase of $L(s)$ are illustrated in Figure 2. The amplitude curve is a straight line of constant slope -20α dB/dec, and the phase curve is a horizontal line positioned at $-\alpha\pi/2$ rad. The Nyquist curve is simply the straight line through the origin, $\arg L(j\omega) = -\alpha\pi/2$ rad.

This choice of $L(s)$ gives a closed-loop system with the desirable property of being insensitive to gain changes. If the gain changes, the crossover frequency ω_c will change, but the phase margin of the system remains $\mathrm{PM} = \pi(1-\alpha/2)$ rad, independent of the value of the gain. This can be seen from the curves of amplitude and phase of Figure 2.

The closed-loop transfer function of fractional-order control system of Figure 1 is given by

$$G(s) = \frac{L(s)}{1+L(s)} = \frac{1}{(s/\omega_c)^{\alpha}+1}, \quad 1 < \alpha < 2. \tag{2.2}$$

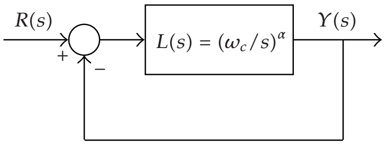


Figure 1: Fractional-order control system with open-loop transfer function $L(s)$.

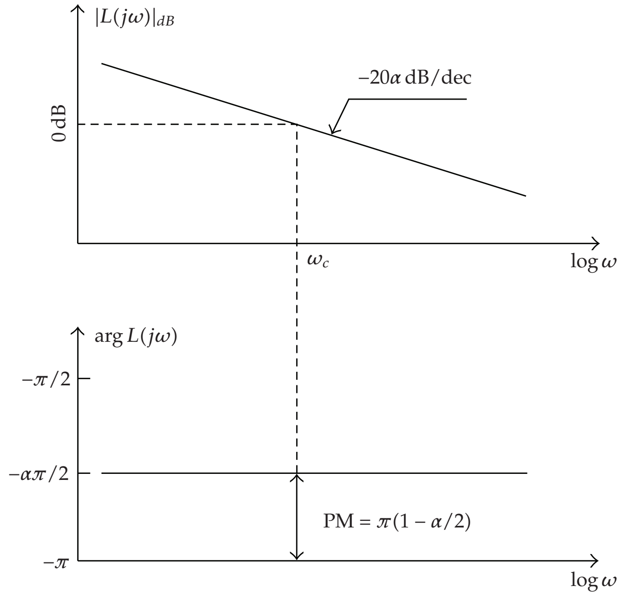


Figure 2: Bode diagrams of amplitude and phase of $L(j\omega)$ for $1 < \alpha < 2$.

The unit step response of $G(s)$ is given by the expression:

$$y_d(t) = L^{-1}\left\{\frac{1}{s}G(s)\right\} = L^{-1}\left\{\frac{\omega_c^\alpha}{s(s^\alpha + \omega_c^\alpha)}\right\} = 1 - \sum_{n=0}^{\infty}\frac{\left[-(\omega_c t)^\alpha\right]^n}{\Gamma(1+\alpha n)} = 1 - E_\alpha\left[-(\omega_c t)^\alpha\right]. \tag{2.3}$$

For the tuning of PID controllers, we address the fractional-order transfer function (2.2) as the reference system [8]. With the order α and the crossover frequency ω_c we can establish the overshoot and the speed of the output response, respectively. For that purpose we consider the closed-loop system shown in Figure 3, where $G_c(s)$ and $G_p(s)$ are the PID controller and the plant transfer functions, respectively.

The transfer function of the PID controller is

$$G_c(s) = \frac{U(s)}{E(s)} = K\left(1 + \frac{1}{T_i s} + T_d s\right), \tag{2.4}$$

where $E(s)$ is the error signal and $U(s)$ is the controller's output. The parameters K, T_i, and T_d are the proportional gain, the integral time constant, and the derivative time constant of the controller, respectively.

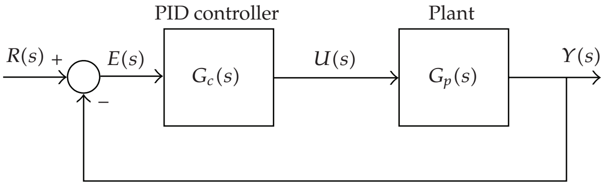


Figure 3: Closed-loop control system with PID controller $G_c(s)$.

The design of the PID controller will consist on the determination of the optimum PID set gains (K, T_i, T_d) that minimize J, the integral of the square error (ISE), defined as

$$J = \int_0^{\infty} [y(t) - y_d(t)]^2 dt, \tag{2.5}$$

where $y(t)$ is the step response of the closed-loop system with the PID controller (Figure 3) and $y_d(t)$ is the desired step response of the fractional-order transfer function (2.2) given by (2.3).

To illustrate the effectiveness of proposed methodology we consider the third-order plant transfer function:

$$G_p(s) = \frac{K_p}{(s+1)^3} \tag{2.6}$$

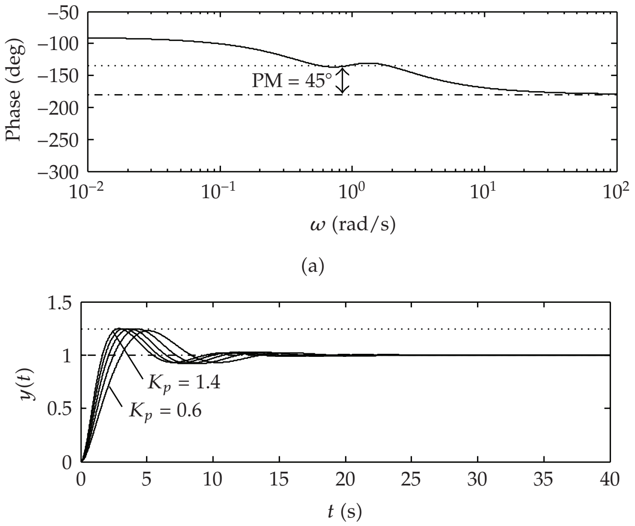
with nominal gain $K_p = 1$.

Figure 4 shows the step responses and the Bode diagrams of phase of the closed-loop system with the PID for the transfer function $G_p(s)$ for gain variations around the nominal gain ($K_p = 1$) corresponding to $K_p = \{0.6, 0.8, 1.0, 1.2, 1.4\}$, that is, for a variation up to ±40% of its nominal value. The system was tuned for $\alpha = 3/2$ (PM = 45°), $\omega_c = 0.8\,\text{rad/s}$. We verify that we get the same desired iso-damping property corresponding to the prescribed (α, ω_c) values.

In fact, we observe that the step responses have an almost constant overshoot independent of the variation of the plant gain around the gain crossover frequency ω_c. Therefore, the proposed methodology is capable of producing closed-loop systems robust to gain variations and step responses exhibiting an iso-damping property. The proposed method was tested on several systems revealing good results. It was also compared with other tuning methods showing comparable or superior results [8].

3. Fractional PD$^{\alpha}$ Control of a Hexapod Robot

Walking machines allow locomotion in terrain inaccessible to other type of vehicles, since they do not need a continuous support surface, but at the cost of higher requirements for leg coordination and control. For these robots, joint level control is usually implemented through a PID-like scheme with position feedback. Recently, the application of the theory of FC to robotics revealed promising aspects for future developments [9]. With these facts in mind, this section compares different Fractional PD$^{\alpha}$ robot controller tuning, applied to the joint



(a)

(b)

Figure 4: Bode phase diagrams and step responses for the closed-loop system with a PID controller for $G_p(s)$. The PID parameters are $K = 1.9158$, $T_i = 1.1407$, and $T_d = 0.9040$.

control of a walking system (Figure 5) with $n = 6$ legs, equally distributed along both sides of the robot body, having each three rotational joints (i.e., $j = \{1, 2, 3\} \equiv \{\text{hip}, \text{knee}, \text{ankle}\}$) [10].

During this study leg joint $j = 3$ can be either mechanical actuated or motor actuated (Figure 5). For the mechanical actuated case, we suppose that there is a rotational pre-tensioned spring-dashpot system connecting leg links L_{i2} and L_{i3}. This mechanical impedance maintains the angle between the two links while imposing a joint torque [10].

Figure 5 presents the dynamic model for the hexapod body and foot-ground interaction. It is considered robot body compliance because walking animals have a spine that allows supporting the locomotion with improved stability. The robot body is divided in n identical segments (each with mass $M_b n^{-1}$) and a linear spring-damper system (with parameters defined so that the body behaviour is similar to the one expected to occur on an animal) is adopted to implement the intrabody compliance [10]. The contact of the ith robot feet with the ground is modelled through a nonlinear system [11], being the values for the parameters based on the studies of soil mechanics [11].

The general control architecture of the hexapod robot is presented in Figure 6 [12]. In this study we evaluate the effect of different PD$^{\alpha}$, $\alpha \in \Re$, controller implementations for $G_{c1}(s)$, while G_{c2} is a proportional controller with gain $Kp_j = 0.9$ ($j = 1, 2, 3$). For the PD$^{\alpha}$ algorithm, implemented through a discrete-time 4th-order Padé approximation (a_{ij}, $b_{ij} \in \Re$, $j = 1, 2, 3$), we have

$$G_{c1j}(z) \approx Kp_j + K\alpha_j \frac{\sum_{i=0}^{i=u} a_{ij}\ z^{-i}}{\sum_{i=0}^{i=u} b_{ij}\ z^{-i}}, \tag{3.1}$$

where Kp_j and $K\alpha_j$ are the proportional and derivative gains, respectively, and α_j is the fractional order, for joint j. Therefore, the classical PD^1 algorithm occurs when the fractional order $\alpha_j = 1.0$.

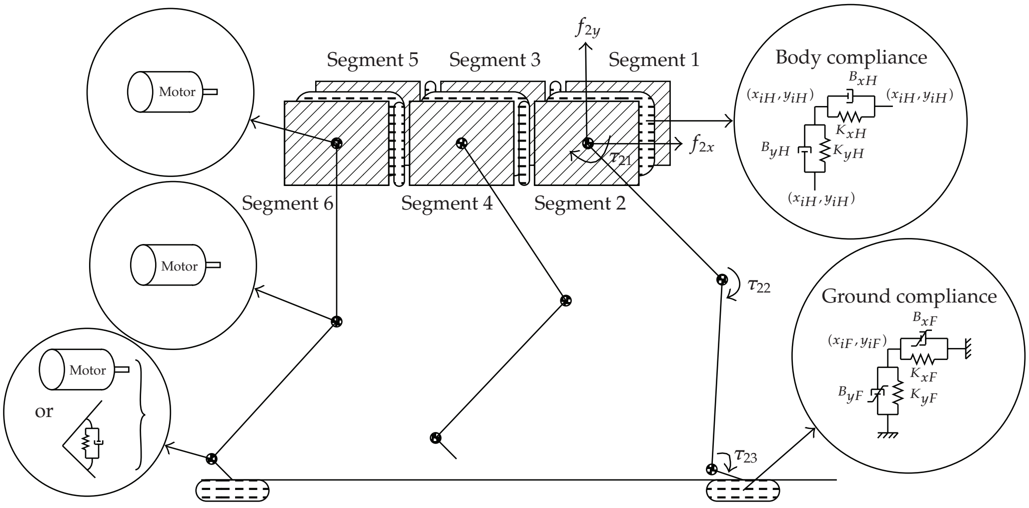


Figure 5: Model of the robot body and foot-ground interaction.

It is analysed the system performance of the different PD$^{\alpha}$ tuning, during a periodic wave gait at a constant forward velocity V_F, for two cases: two leg joints are motor actuated and the ankle joint is mechanical actuated and the three leg joints are fully motor actuated [10].

The analysis is based on the formulation of two indices measuring the mean absolute density of energy per traveled distance (E_{av}) and the hip trajectory errors (ε_{xyH}) during walking, according to

$$\begin{gathered}
E_{av} = \frac{1}{d}\sum_{i=1}^{n}\sum_{j=1}^{m}\int_0^T \left|\boldsymbol{\tau}_{ij}(t)\dot{\boldsymbol{\theta}}_{ij}(t)\right| dt \quad \left[\mathbf{Jm}^{-1}\right], \\
\varepsilon_{xyH} = \sum_{i=1}^{n}\sqrt{\frac{1}{N_s}\sum_{k=1}^{N_s}\left({\Delta_{ixH}}^2 + {\Delta_{iyH}}^2\right)} \quad [\mathbf{m}], \\
\Delta_{ixH} = x_{iHd}(k) - x_{iH}(k), \qquad \Delta_{iyH} = y_{iHd}(k) - y_{iH}(k).
\end{gathered} \tag{3.2}$$

To tune the different controller implementations we adopt a systematic method, testing and evaluating several possible combinations of parameters, for all controller implementations. Therefore, we adopt the $G_{c1}(s)$ parameters that establish a compromise in what concerns the simultaneous minimisation of E_{av} and ε_{xyH}. Moreover, it is assumed high-performance joint actuators, with a maximum actuator torque of $\tau_{ij\text{Max}} = 400$ Nm, and the desired angle between the foot and the ground (assumed horizontal) is established as $\theta_{i3hd} = -15^{\circ}$. We tune the PD$^{\alpha}$ joint controllers for different values of the fractional order α_j while making $\alpha_1 = \alpha_2 = \alpha_3$.

We start by considering that leg joints 1 and 2 are motor actuated and joint 3 is mechanical actuated. For this case we tune the PD$^{\alpha}$ joint controllers for different values of the fractional order α_j, with step $\Delta\alpha_j = 0.1$, namely, $\alpha_j = \{-0.9, -0.8, \ldots, +0.9\}$. Afterwards,

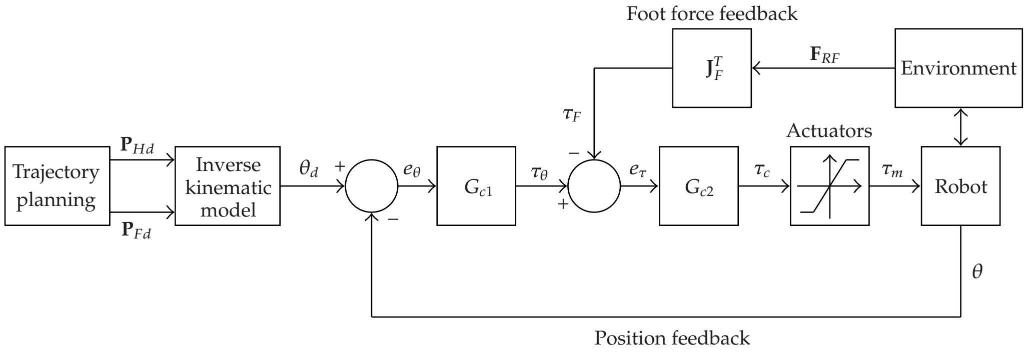

Foot force feedback
$\mathbf{J}_F^T$
$\mathbf{F}_{RF}$
Environment
τ_F
Actuators
Trajectory planning
$\mathbf{P}_{Hd}$
$\mathbf{P}_{Fd}$
Inverse kinematic model
θ_d
+
−
e_θ
G_{c1}
τ_θ
−
+
e_τ
G_{c2}
τ_c
τ_m
Robot
θ
Position feedback

Figure 6: Hexapod robot control architecture.

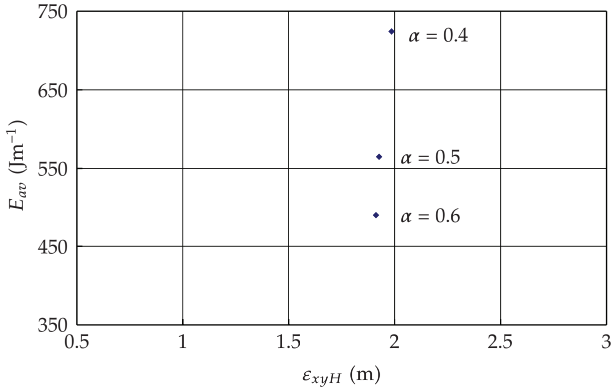


Figure 7: Locus of E_{av} versus ε_{xyH} for the different values of α in the $G_{c1}(s)$ tuning, when establishing a compromise between the minimisation of E_{av} and ε_{xyH}, with $G_{c2} = 0.9$, joints 1 and 2 motor actuated and joint 3 mechanical actuated.

we consider that joint 3 is also motor actuated, and we repeat the controller tuning procedure versus α_j.

For the first situation under study, we verify that the value of $\alpha_j = 0.6$ (Figure 7), being the gains of the PD$^\alpha$ controller $K_{p1} = 2500$, $K_{\alpha 1} = 800$, $K_{p2} = 300$, $K_{\alpha 2} = 100$ and the parameters of the mechanical spring-dashpot system for the ankle actuation $K_3 = 1$, $B_3 = 2$, presents the best compromise situation between the simultaneous minimisation of ε_{xyH} and E_{av}.

Regarding the case when all joints are motor actuated, Figure 8 presents the best controller tuning for different values of α_j. The experiments reveal the superior performance of the PD$^\alpha$ controller for $\alpha_j \approx 0.5$, with $K_{p1} = 15000$, $K_{\alpha 1} = 7200$, $K_{p2} = 1000$, $K_{\alpha 2} = 800$, and $K_{p3} = 150$, $K_{\alpha 3} = 240$.

For $\alpha_j = \{0.1, 0.2, 0.3, 0.4\}$ the results are very poor and for $\alpha_j = \{-0.9, \ldots, -0.1\} \cup \{+0.9\}$, the hexapod locomotion is unstable. Furthermore, we conclude that the best case corresponds to all leg joints being motor actuated.

In conclusion, the experiments reveal the superior performance of the FO controller for $\alpha_j \approx 0.5$ and a robot with all motor actuated joints, as can be concluded analysing the curves for the joint actuation torques τ_{1jm} (Figure 9) and for the hip trajectory tracking errors Δ_{1xH} and Δ_{1yH} (Figure 10).

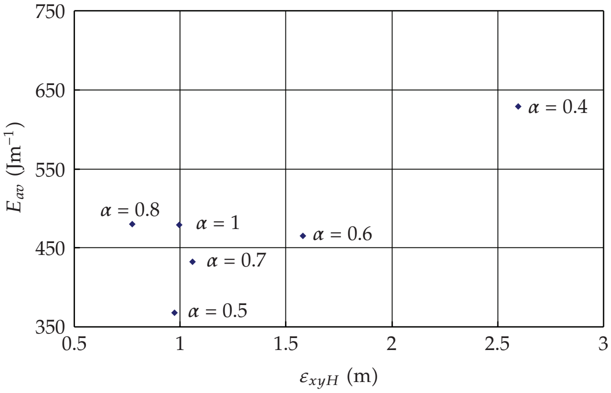
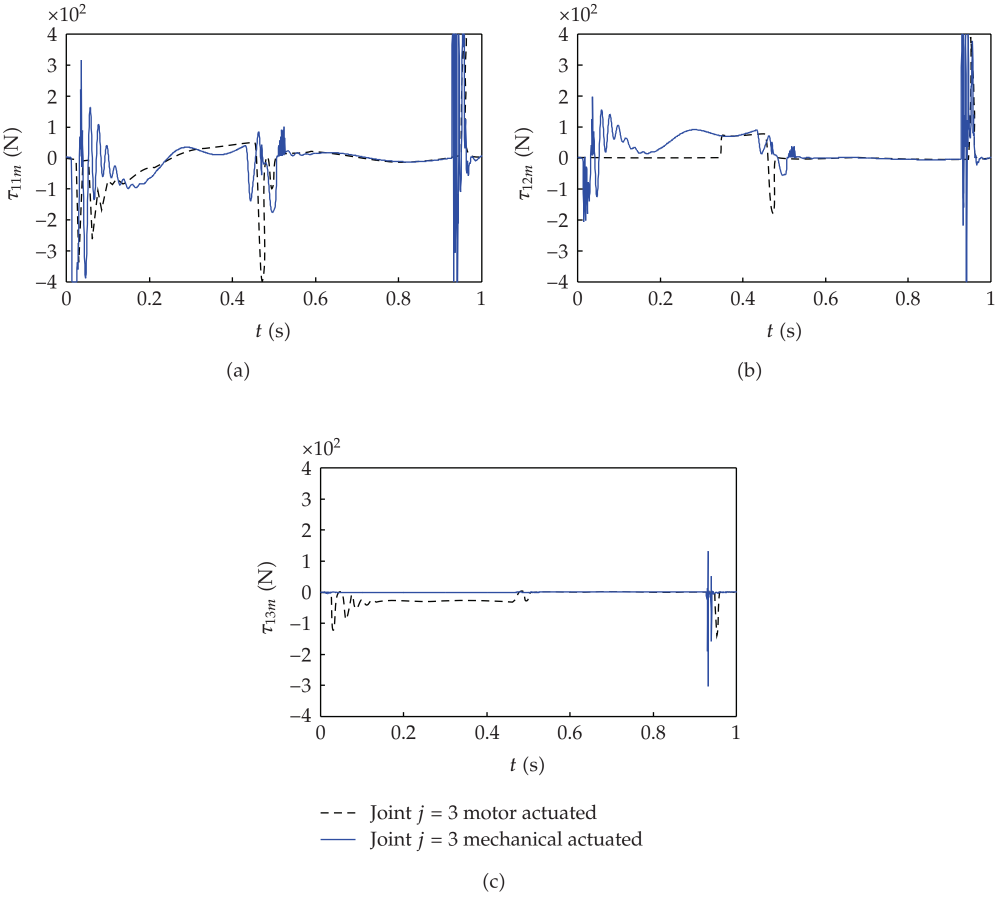


Figure 8: Locus of E_{av} versus ε_{xyH} for the different values of α in the $G_{c1}(s)$ tuning, when establishing a compromise between the minimisation of E_{av} and ε_{xyH}, with $G_{c2} = 0.9$ and all joints motor actuated.

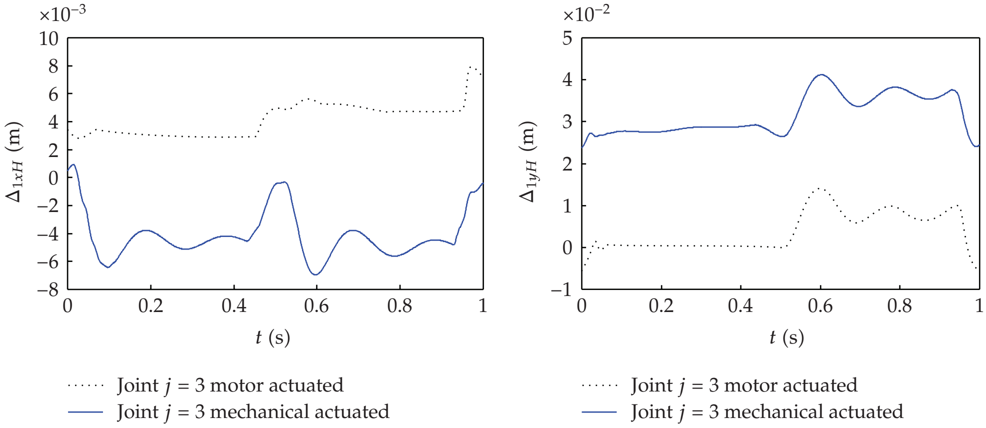


(a)

(b)

(c)

Figure 9: Plots of τ_{1jm} versus t, with joints 1 and 2 motor actuated and joint 3 mechanical actuated and all joints motor actuated, for $\alpha_j = 0.5$.



(a) (b)

Figure 10: Plots of Δ_{1xH} and Δ_{1yH} versus t, with joints 1 and 2 motor actuated and joint 3 mechanical actuated and all joints motor actuated, for $\alpha_j = 0.5$.

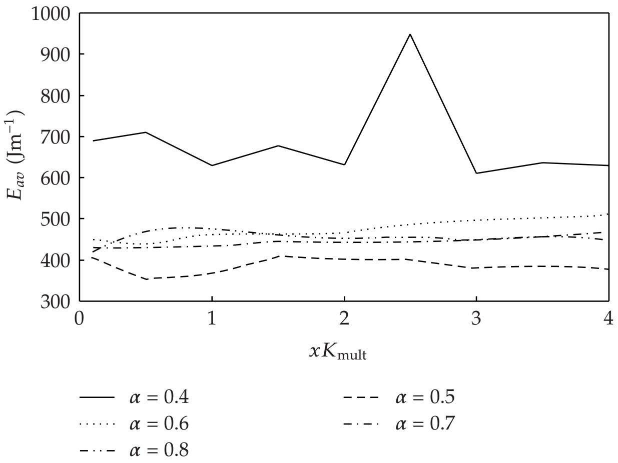


Figure 11: Performance index E_{av} versus K_{mult} for the $G_{c1}(s)$ PD$^{\alpha}$ controller tuning having all joints motor actuated.

Since the objective of the walking robots is to walk in natural terrains, in the sequel it is examined how the different controller tunings behave under different ground properties, considering that all joints are motor actuated. For this case, and considering the previously tuning controller parameters, the values of $\{K_{xF}, B_{xF}, K_{yF}, B_{yF}\}$ are varied simultaneously through a multiplying factor K_{mult} that is varied in the range $[0.1, 4.0]$. This variation for the ground model parameters allows the simulation of the ground behaviour for growing stiffness, from peat to gravel [11].

The performance measure E_{av} versus the multiplying factor of the ground parameters K_{mult} is presented on Figure 11. Analysing the system performance from the viewpoint of the index E_{av}, it is possible to conclude that the best PD$^{\alpha}$ implementation occurs for the fractional order $\alpha_j = 0.5$. Moreover, it is clear that the performances of the different controller implementations are almost constant on all range of the ground parameters, with the exception of the fractional order $\alpha_j = 0.4$. For this case, E_{av} presents a significant variation with K_{mult}. Therefore, we conclude that the controller responses are quite similar, meaning that these algorithms are robust to variations of the ground characteristics [12].

4. Fractional Dynamics in the Trajectory Control of Redundant Manipulators

A redundant manipulator is a robotic arm possessing more degrees of freedom (*dof*) than those required to establish an arbitrary position and orientation of the end effector. Redundant manipulators offer several potential advantages over non-redundant arms. In a workspace with obstacles, the extra degrees of freedom can be used to move around or between obstacles and thereby to manipulate in situations that otherwise would be inaccessible [20–23].

When a manipulator is redundant, it is anticipated that the inverse kinematics admits an infinite number of solutions. This implies that, for a given location of the manipulator's gripper, it is possible to induce a self-motion of the structure without changing the location of the end effecter. Therefore, the arm can be reconfigured to find better postures for an assigned set of task requirements.

Several kinematic techniques for redundant manipulators control the gripper through the rates at which the joints are driven, using the pseudoinverse of the Jacobian [22, 24]. Nevertheless, these algorithms lead to a kind of chaotic motion with unpredictable arm configurations.

Having these ideas in mind, Section 4.1 introduces the fundamental issues for the kinematics of redundant manipulators. Based on these concepts, Section 4.2 presents the trajectory control of a three *dof* robot. The results reveal a chaotic behavior that is further analyzed in Section 4.3.

4.1. Kinematics of Redundant Manipulators

A kinematically redundant manipulator has more *dof* than those required to define an arbitrary position and orientation of the gripper. In Figure 12 is depicted a planar manipulator with $k \in \aleph$ rotational (R) joints that is redundant for $k > 2$. When a manipulator is redundant it is anticipated that the inverse kinematics admits an infinite number of solutions. This implies that, for a given location of the manipulator's gripper, it is possible to induce a self-motion of the structure without changing the location of the gripper. Therefore, redundant manipulators can be reconfigured to find better postures for an assigned set of task requirements but, on the other hand, have a more complex structure requiring adequate control algorithms.

We consider a manipulator with n degrees of freedom whose joint variables are denoted by $\mathbf{q} = [q_1, q_2, \ldots, q_n]^T$. We assume that a class of tasks, we are interested in can be described by m variables, $\mathbf{x} = [x_1, x_2, \ldots, x_m]^T (m < n)$ and that the relation between $\mathbf{q}$ and $\mathbf{x}$ is given by

$$\mathbf{x} = f(\mathbf{q}), \tag{4.1}$$

where f is a function representing the direct kinematics.

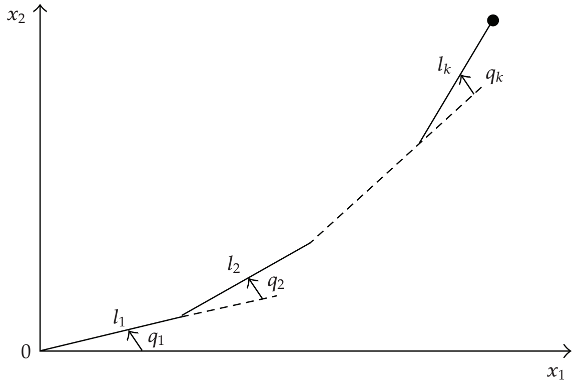


Figure 12: A planar redundant planar manipulator with k rotational joints.

Differentiating (4.1) with respect to time yields

$$\dot{\mathbf{x}} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}, \tag{4.2}$$

where $\dot{\mathbf{x}} \in \Re^m$, $\dot{\mathbf{q}} \in \Re^n$, and $\mathbf{J}(\mathbf{q}) = \partial f(\mathbf{q})/\partial \mathbf{q} \in \Re^{m \times n}$. Hence, it is possible to calculate a path $\mathbf{q}(t)$ in terms of a prescribed trajectory $\mathbf{x}(t)$ in the operational space. We assume that the following condition is satisfied:

$$\max \operatorname{rank}\{\mathbf{J}(\mathbf{q})\} = m. \tag{4.3}$$

Failing to satisfy this condition usually means that the selection of manipulation variables is redundant and the number of these variables m can be reduced. When condition (4.3) is verified, we say that the degree of redundancy of the manipulator is $n-m$. If, for some $\mathbf{q}$ we have

$$\operatorname{rank}\{\mathbf{J}(\mathbf{q})\} < m \tag{4.4}$$

then the manipulator is in a singular state. This state is not desirable because, in this region of the trajectory, the manipulating ability is very limited.

Many approaches for solving redundancy [25, 26] are based on the inversion of (4.2). A solution in terms of the joint velocities is sought as

$$\dot{\mathbf{q}} = \mathbf{J}^{\#}(\mathbf{q})\dot{\mathbf{x}}, \tag{4.5}$$

where $J^{\#}$ is one of the generalized inverses of the $\mathbf{J}$ [26–28]. It can be easily shown that a more general solution to (4.2) is given by

$$\dot{\mathbf{q}} = \mathbf{J}^{+}(\mathbf{q})\dot{\mathbf{x}} + [\mathbf{I} - \mathbf{J}^{+}(\mathbf{q})\mathbf{J}(\mathbf{q})]\dot{\mathbf{q}}_0, \tag{4.6}$$

where $\mathbf{I}$ is the $n \times n$ identity matrix and $\dot{\mathbf{q}}_0 \in \Re^n$ is a $n \times 1$ arbitrary joint velocity vector and $\mathbf{J}^{+}$ is the pseudoinverse of the $\mathbf{J}$. The solution (4.6) is composed of two terms. The

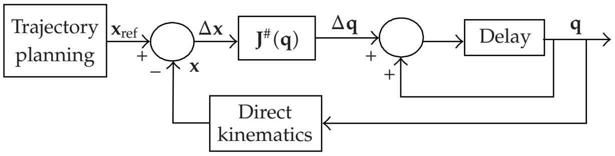


Figure 13: Block diagram of the closed-loop inverse kinematics algorithm with the pseudoinverse.

first term is relative to minimum norm joint velocities. The second term, the *homogeneous solution*, attempts to satisfy the additional constraints specified by $\dot{\mathbf{q}}_0$. Moreover, the matrix $\mathbf{I} - \mathbf{J}^{+}(\mathbf{q})\mathbf{J}(\mathbf{q})$ allows the projection of $\dot{\mathbf{q}}_0$ in the null space of $\mathbf{J}$. A direct consequence is that it is possible to generate internal motions that reconfigure the manipulator structure without changing the gripper position and orientation [27–30]. Another aspect revealed by the solution of (4.6) is that repetitive trajectories in the operational space do not lead to periodic trajectories in the joint space. This is an obstacle for the solution of many tasks because the resultant robot configurations have similarities with those of a chaotic system.

4.2. Robot Trajectory Control

The direct kinematics and the Jacobian of a 3-link planar manipulator with rotational joints (3*R* robot) has a simple recursive nature according with the expressions:

$$\begin{gathered}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} l_1 C_1 + l_2 C_{12} + l_3 C_{123} \\ l_1 S_1 + l_2 S_{12} + l_3 S_{123} \end{bmatrix}, \\ \mathbf{J} = \begin{bmatrix} -l_1 S_1 - \cdots - l_3 S_{123} & \cdots & -l_3 S_{123} \\ l_1 C_1 + \cdots + l_3 C_{123} & \cdots & l_3 C_{123} \end{bmatrix}, \end{gathered} \tag{4.7}$$

where l_i is the length of link i, $q_{i\ldots k} = q_i + \cdots + q_k$, $S_{i\ldots k} = \mathrm{Sin}(q_{i\ldots k})$, and $C_{i\ldots k} = \mathrm{Cos}(q_{i\ldots k})$.

During all the experiments it is considered $\Delta t = 10^{-3}$ seconds, $L_{\mathrm{TOT}} = l_1 + l_2 + l_3 = 3$, and $l_1 = l_2 = l_3$.

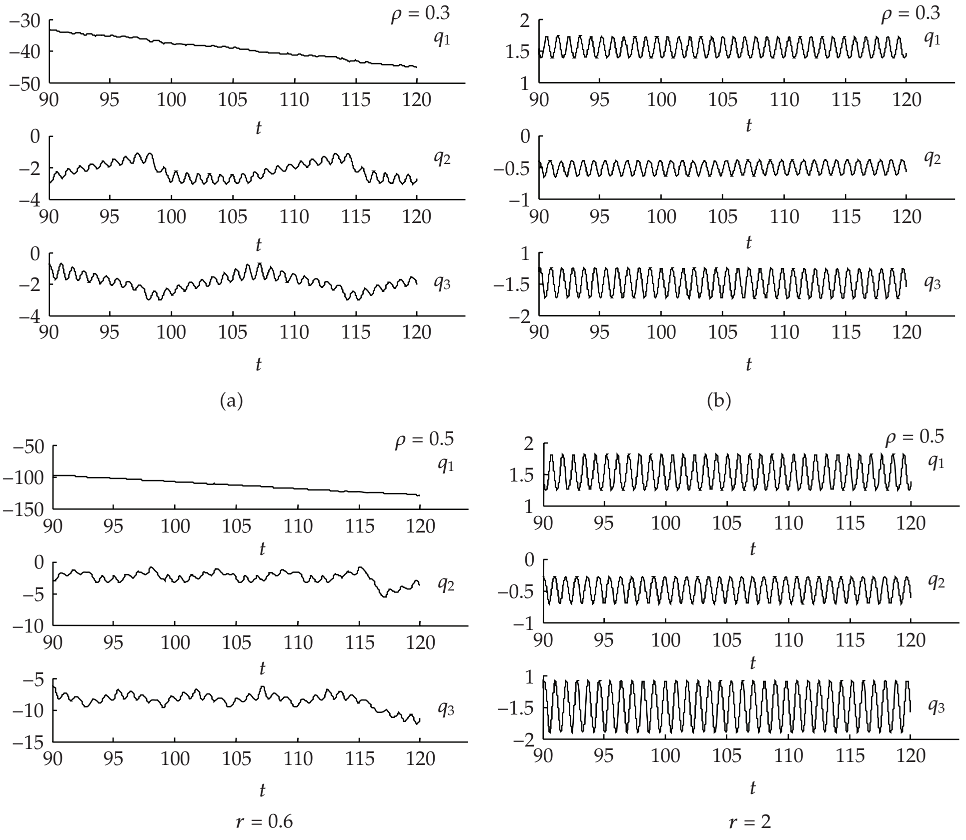
In the closed-loop pseudoinverse's method the joint positions can be computed through the time integration of the velocities according with the block diagram of the inverse kinematics algorithm depicted in Figure 13, where $\mathbf{x}_{\mathbf{ref}}$ represents the vector of reference coordinates of the robot gripper in the operational space.

Based on (4.7) we analyze the kinematic performances of the 3*R*-robot when repeating a circular motion in the operational space with frequency $\omega_0 = 7.0\,\mathrm{rad\,s^{-1}}$, centre at distance $r = [x_1^2 + x_2^2]^{1/2}$ and radius ρ.

Figure 14 shows the joint positions for the inverse kinematic algorithm (4.5) for $r = \{0.6, 2.0\}$ and $\rho = \{0.3, 0.5\}$. We observe that the following hold.

(i) For $r = 0.6$ occur unpredictable motions with severe variations that lead to high joint transients [13]. Moreover, we verify a low-frequency signal modulation that depends on the circle being executed.

(ii) For $r = 2.0$ the motion is periodic with frequency identical to $\omega_0 = 7.0\,\mathrm{rad\,s^{-1}}$.



(a) (b)

(c) (d)

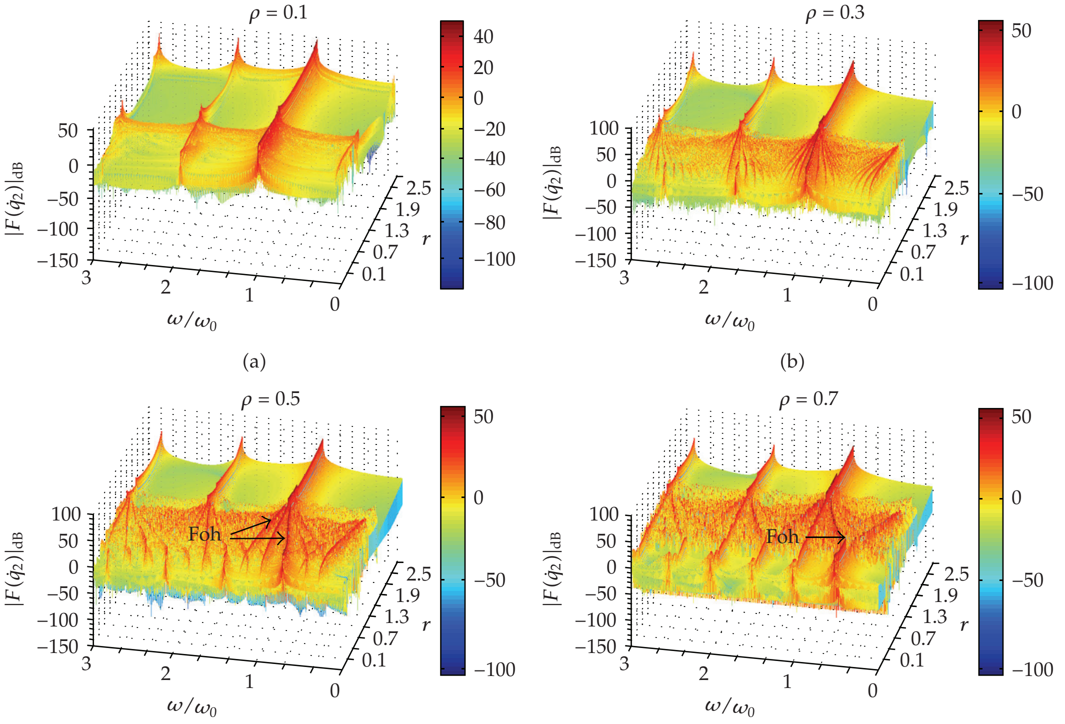
Figure 14: The $3R$-robot joint positions versus time using the pseudoinverse method for $r = \{0.6, 2.0\}$ and $\rho = \{0.3, 0.5\}$.

4.3. Analysis of the Robot Trajectories

In the previous subsection we verified that the pseudoinverse-based algorithm leads to unpredictable arm configurations. In order to gain further insight into the pseudoinverse nature several distinct experiments are devised in the sequel during a time window of 300 cycles. Therefore, in a first set of experiments we calculate the Fourier transform of the $3R$-robot joints velocities for a circular repetitive motion with frequency $\omega_0 = 7.0\,\mathrm{rad\,s^{-1}}$, radius $\rho = \{0.1, 0.3, 0.5, 0.7\}$, and radial distances $r \in]0, L_{\mathrm{TOT}} - \rho[$.

Figure 15 shows $|F\{\dot{q}_2(t)\}|$ versus the frequency ratio ω_0/ω and the distance r where $F\{\ \}$ represents the Fourier operator. Is verified an interesting phenomenon induced by the gripper repetitive motion ω_0 because a large part of the energy is distributed along several subharmonics. These fractional-order harmonics (*foh*) depend on r and ρ making a complex pattern with similarities with those revealed by chaotic systems. Furthermore, we observe the existence of several distinct regions depending on r.

For example, selecting in Figure 15 several distinct cases, namely for $r = \{0.08, 0.30, 0.53, 1.10, 1.30, 2.00\}$, we have the different signal Fourier spectra clearly visible in Figure 16.



(a) (b) (c) (d)

Figure 15: $|F\{\dot{q}_2(t)\}|$ of the $3R$-robot during 300 cycles versus r and ω/ω_0, for $\rho = \{0.1, 0.3, 0.5, 0.7\}$, $\omega_0 = 7.0\,\text{rad s}^{-1}$.

In the authors' best knowledge the *foh* are aspects of fractional dynamics [14, 15, 31], but a final and assertive conclusion about a physical interpretation is a matter still to be explored.

For joints velocities 1 and 3 the results are similar to the verified ones for joint velocity 2.

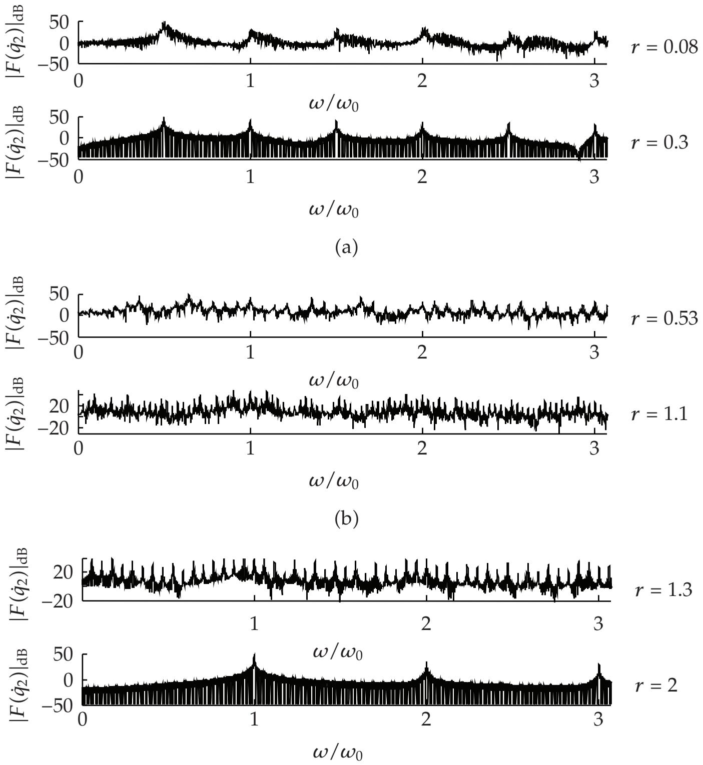
5. Heat Diffusion

The heat diffusion is governed by a linear one-dimensional partial differential equation (PDE) of the form:

$$\frac{\partial u}{\partial t} = k\frac{\partial^2 u}{\partial x^2}, \tag{5.1}$$

where k is the diffusivity, t is the time, u is the temperature, and x is the space coordinate. However, (5.1) involves the solution of a PDE of parabolic type for which the standard theory guarantees the existence of a unique solution [16].

For the case of a planar perfectly isolated surface we usually apply a constant temperature U_0 at $x = 0$ and analyzes the heat diffusion along the horizontal coordinate x.



(a)

(b)

(c)

Figure 16: $|F\{\dot{q}_2(t)\}|$ of the $3R$-robot during 300 cycles versus the frequency ratio ω/ω_0, for $r = \{0.08, 0.30, 0.53, 1.10, 1.30, 2.00\}$, $\rho = 0.7$, $\omega_0 = 7.0\,\mathrm{rad\,s^{-1}}$.

Under these conditions, the heat diffusion phenomenon is described by a noninteger-order model:

$$U(x,s) = \frac{U_0}{s}G(s) \qquad G(s) = e^{-x\sqrt{s/k}}, \tag{5.2}$$

where x is the space coordinate, U_0 is the boundary condition, and $G(s)$ is the system transfer function.

In our study, the simulation of the heat diffusion is performed by adopting the Crank-Nicholson implicit numerical integration based on the discrete approximation to differentiation as [16, 17]

$$-ru[j+1,i+1] + (2+r)u[j+1,i] - ru[j+1,i-1] = ru[j,i+1] + (2-r)u[j,i] + u[j,i-1], \tag{5.3}$$

where $r = \Delta t(\Delta x^2)^{-1}$, $\{\Delta x, \Delta t\}$, and $\{i, j\}$ are the increments and the integration indices for space and time, respectively.

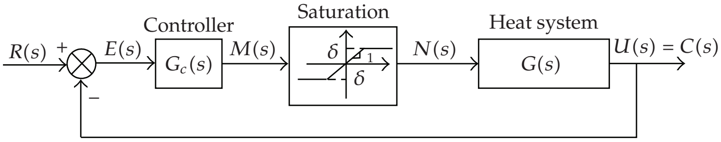


Figure 17: Closed-loop system with PID controller $G_c(s)$.

5.1. Control Strategies

The generalized PID controller $G_c(s)$ has a transfer function of the form

$$G_c(s) = K\left[1 + \frac{1}{T_i s^{\alpha}} + T_d s^{\beta}\right], \tag{5.4}$$

where α and β are the orders of the fractional integrator and differentiator, respectively. The constants K, T_i, and T_d are correspondingly the proportional gain, the integral time constant, and the derivative time constant.

Clearly, taking $(\alpha, \beta) = \{(1,1), (1,0), (0,1), (0,0)\}$ we get the classical $\{\mathrm{PID}, \mathrm{PI}, \mathrm{PD}, \mathrm{P}\}$ controllers, respectively.

The $\mathrm{PI}^{\alpha}\mathrm{D}^{\beta}$ controller is more flexible and gives the possibility of adjusting more carefully the closed-loop system characteristics.

In the following two subsections, we analyze the system of Figure 17 by adopting the classical integer-order PID and a fractional PID^{β}, respectively.

5.2. PID Tuning Using the Ziegler-Nichols Rule

In this subsection, we analyze the closed-loop system with a conventional PID controller given by the transfer function (5.4) with $\alpha = \beta = 1$. Usually, the PID parameters (K, T_i, T_d) are tuned by using the so-called Ziegler-Nichols open loop (ZNOL) method [17]. The ZNOL heuristics are based on the approximate first-order plus dead-time model:

$$\widehat{G}(s) = \frac{K_p}{\tau s + 1} e^{-sT}. \tag{5.5}$$

For the heat system, the resulting parameters are $\{K_p, \tau, T\} = \{0.52, 162, 28\}$ leading to the PID constants $\{K, T_i, T_d\} = \{18.07, 34.0, 8.5\}$.

A step input is applied at $x = 0.0\,\mathrm{m}$ and the closed-loop response $c(t)$ is analyzed for $x = 3.0\,\mathrm{m}$, without actuator saturation (Figure 18). We verify that the system with a PID controller, tuned through the ZNOL heuristics, does not produce satisfactory results giving a significant overshoot ov and a large settling time t_s, namely, $\{t_s, t_p, t_r, ov(\%)\} \equiv \{44.8, 27.5, 12.0, 68.56\}$, where t_p represents the peak time and t_r the rise time. We consider

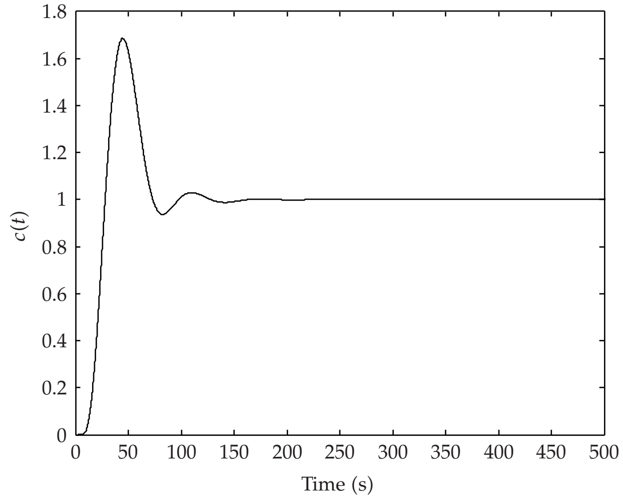


Figure 18: Step responses of the closed-loop system for the PID controller and $x = 3.0\,\mathrm{m}$.

two indices that measure the response error, namely, the integral square error (ISE) and the integral time square error (ITSE) criteria defined as

$$\begin{aligned}\mathrm{ISE} &= \int_0^\infty [r(t) - c(t)]^2 dt, \\ \mathrm{ITSE} &= \int_0^\infty t[r(t) - c(t)]^2 dt.\end{aligned} \tag{5.6}$$

We can use other performance criteria such as the integral absolute error (IAE) or the integral time absolute error (ITAE); however, in the present case, the ISE and the ITSE criteria have produced the best results and are adopted in the study.

In this case, the ZNOL PID tuning leads to the values $(\mathrm{ISE}, \mathrm{ITSE}) = (27.53, 613.97)$. The poor results indicate again that the method of tuning may not be the most adequate for the control of the heat system.

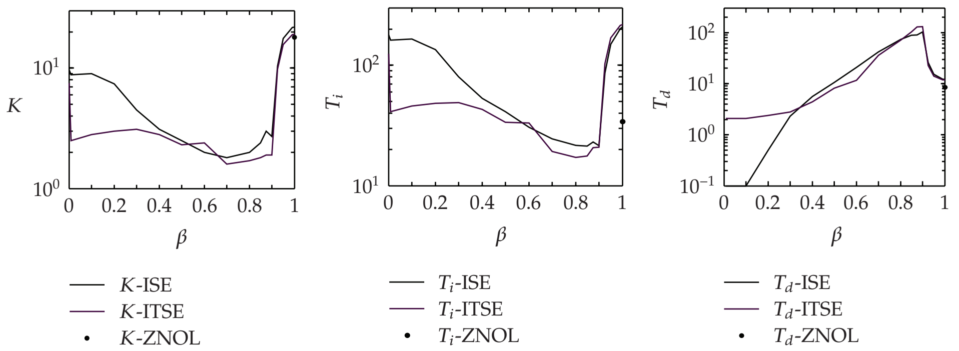
In fact, the inherent fractional dynamics of the system lead us to consider other configurations. In this perspective, we propose the use of fractional controllers tuned by the minimization of the indices ISE and ITSE.

5.3. *PID$^\beta$ Tuning Using Optimization Indices*

In this subsection, we analyze the closed-loop system under the action of the PID$^\beta$ controller given by the transfer function (5.4) with $\alpha = 1$ and $0 \le \beta \le 1$. The fractional derivative term $T_d s^\beta$ in (5.4) is implemented through a fourth-order Padé discrete rational transfer function. It used a sampling period of $T = 0.1$ second.

The PID$^\beta$ controller is tuned by the minimization of an integral performance index. For that purpose, we adopt the ISE and ITSE criteria.

A step reference input $R(s) = 1/s$ is applied at $x = 0.0\,\mathrm{m}$ and the output $c(t)$ is analyzed for $x = 3.0\,\mathrm{m}$, without actuator saturation. The heat system is simulated for 3000



(a) (b) (c)

Figure 19: The PID$^\beta$ parameters (K, T_i, T_d) versus β for the ISE and ITSE optimization criteria. The dot represents the PID-ZNOL.

seconds. Figure 19 illustrates the variation of the fractional PID parameters (K, T_i, T_d) as function of the order's derivative β, for the ISE and the ITSE criteria. The dots represent the values corresponding to the classical PID (ZNOL-tuning) addressed in the previous section.

The curves reveal that for $\beta < 0.4$ the parameters (K, T_i, T_d) are slightly different, for the two ISE and ITSE criteria, while for $\beta \geq 0.4$ they lead to almost similar values. This fact indicates a large influence of a weak-order derivative on system's dynamics.

To further illustrate the performance of the fractional-order controllers a saturation nonlinearity is included in the closed-loop system of Figure 17 and inserted in series with the output of the controller $G_c(s)$. The saturation element is defined as

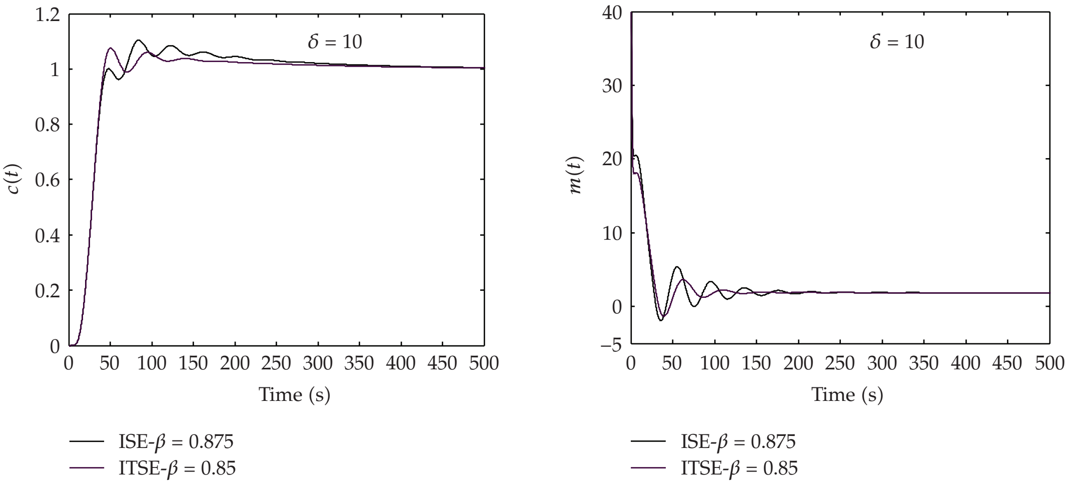
$$n(m) = \begin{cases} m, & |m| < \delta, \\ \delta \operatorname{sign}(m), & |m| \geq \delta. \end{cases} \tag{5.7}$$

The controller performance is evaluated for $\delta = \{20, \ldots, 100\}$ and $\delta = \infty$ which corresponds to a system without saturation. We use the same fractional-PID parameters obtained without considering the saturation nonlinearity.

Figures 20 and 21 show the step responses of the closed-loop system and the corresponding controller output, for the PID$^\beta$ tuned in the ISE and ITSE perspectives for $\delta = 10$ and $\delta = \infty$, respectively. The controller parameters $\{K, T_i, T_d, \beta\}$ correspond to the minimization of those indices leading to the values ISE: $\{K, T_i, T_d, \beta\} \equiv \{3, 23, 90.6, 0.875\}$ and ITSE: $\{K, T_i, T_d, \beta\} \equiv \{1.8, 17.6, 103.6, 0.85\}$.

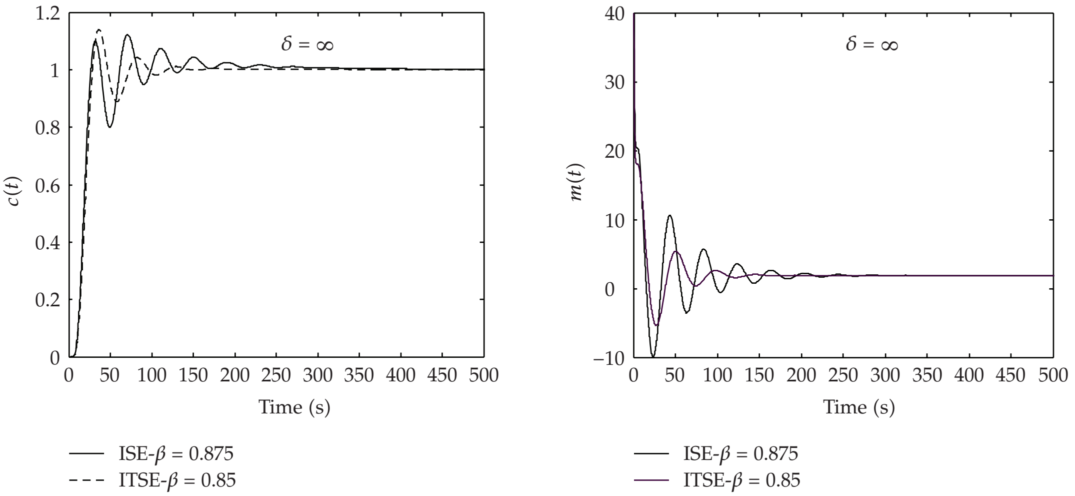
The step responses reveal a large diminishing of the overshoot and the rise time when compared with the integer PID, showing a good transient response and a zero steady-state error. The PID$^\beta$ leads to better results than the classical PID controller tuned through the ZNOL rule. These results demonstrate the effectiveness of the fractional algorithms when used for the control of fractional-order systems. The step response and the controller output are also improved when the saturation level δ is diminished.

Figure 22 depicts the ISE and ITSE indices for $0 \leq \beta \leq 1$, when $\delta = \{20, \ldots, 100\}$ and $\delta = \infty$. We verify the existence of a minimum for $\beta = 0.875$ and $\beta = 0.85$ for the ISE and ITSE cases, respectively. Furthermore, the higher the δ the lower the value of the index.



(a) (b)

Figure 20: Step responses of the closed-loop system and the controller output for the ISE and the ITSE indices, with a PID$^{\beta}$ controller, $\delta = 10$ and $x = 3.0\,\text{m}$.

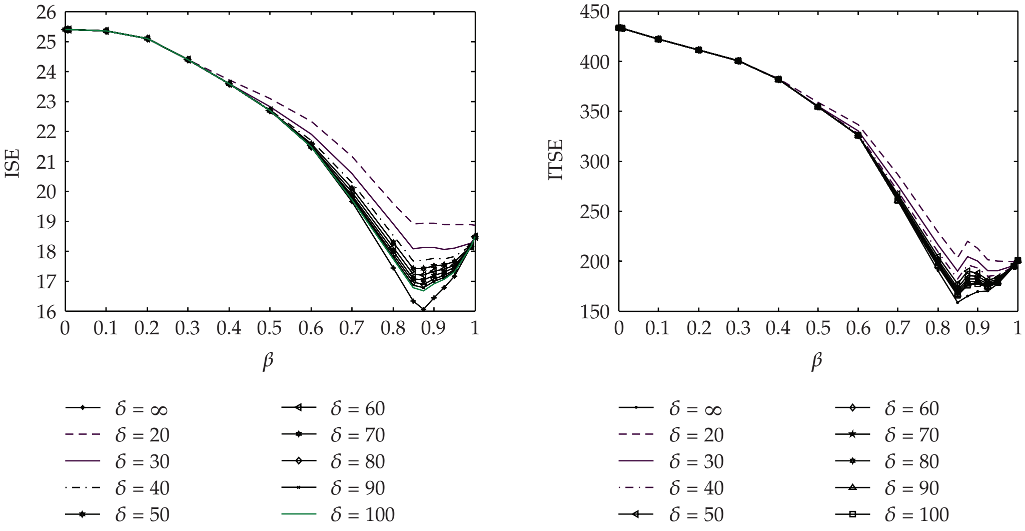


(a) (b)

Figure 21: Step responses of the closed-loop system and the controller output for the ISE and the ITSE indices, with a PID$^{\beta}$ controller, $\delta = \infty$ and $x = 3.0\,\text{m}$.

Figures 23 and 24 show the variation of the settling time t_s, the peak time t_p, the rise time t_r, and the percent overshoot $ov(\%)$, for the closed-loop response tuned through the minimization of the ISE and the ITSE indices, respectively.

In the ISE case t_s, t_p, and t_r diminish rapidly for $0 \leq \beta \leq 0.875$, while for $\beta > 0.875$ the parameters increase smoothly. For the ITSE, we verify the same behavior for $\beta = 0.85$. On the other hand, $ov(\%)$ increases smoothly for $0 \leq \beta \leq 0.7$, while for $\beta > 0.7$ it decreases very quickly, both for the ISE and the ITSE indices.



(a) (b)

Figure 22: ISE and ITSE versus $0 \leq \beta \leq 1$ for $\delta = \{20, \ldots, 100\}$ and $\delta = \infty$.

In conclusion, for $0.85 \leq \beta \leq 0.875$ we get the best controller tuning, superior to the performance revealed by the classical integer-order scheme.

6. Circuit Synthesis Using Evolutionary Algorithms

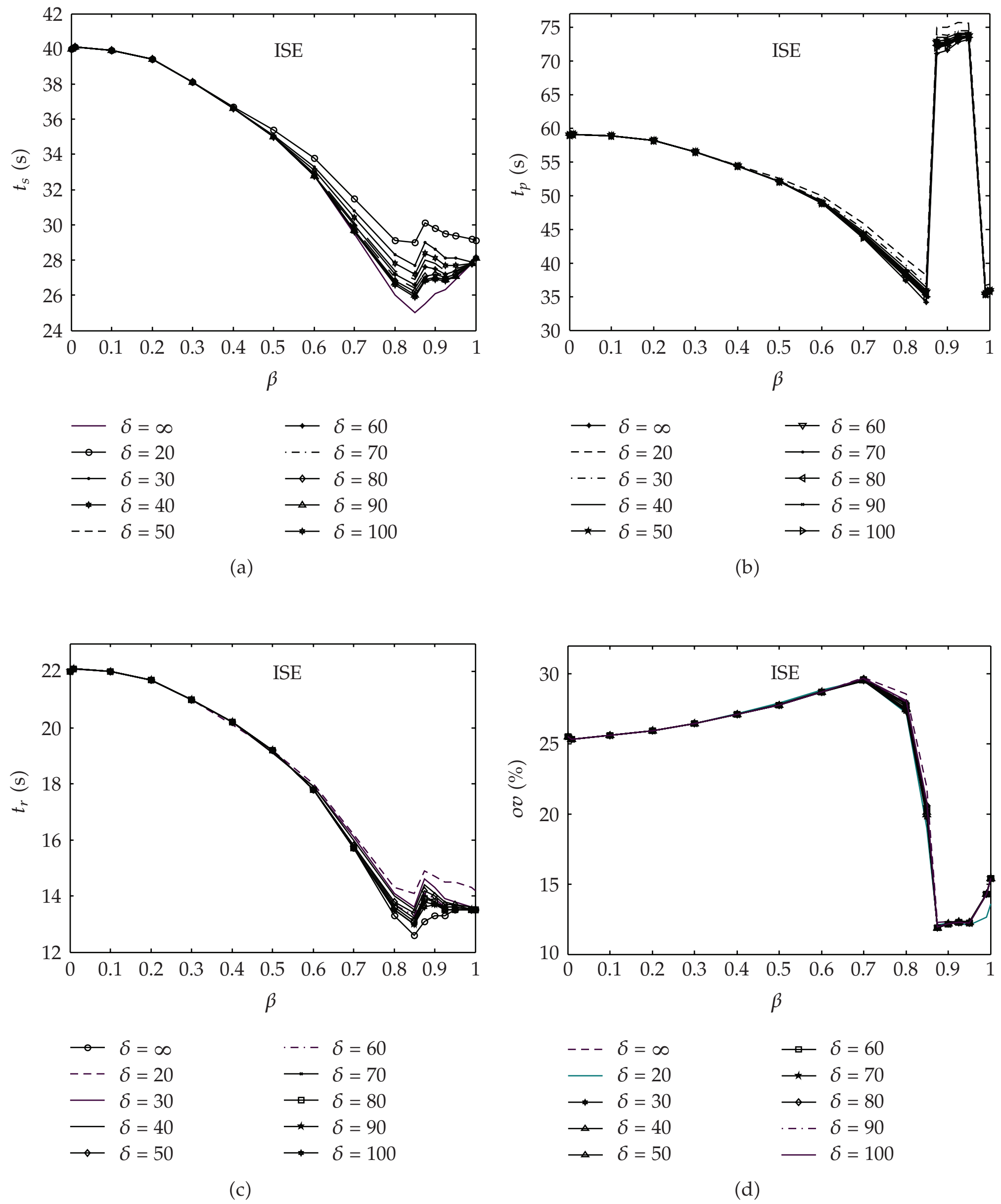
In recent decades evolutionary computation (EC) techniques have been applied to the design of electronic circuits and systems, leading to a novel area of research called Evolutionary Electronics (EE) or Evolvable Hardware (EH). EE considers the concept for automatic design of electronic systems. Instead of using human conceived models, abstractions, and techniques, EE employs search algorithms to develop implementations not achievable with the traditional design schemes, such as the Karnaugh or the Quine-McCluskey Boolean methods.

Several papers proposed designing combinational logic circuits using evolutionary algorithms and, in particular, genetic algorithms (GAs) [32, 33] and hybrid schemes such as the memetic algorithms (MAs) [34].

Particle swarm optimization (PSO) constitutes an alternative evolutionary computation technique, and this paper studies its application to combinational logic circuit synthesis. Bearing these ideas in mind, the organization of this section is as follows. Section 6.1 presents a brief overview of the PSO. Section 6.2 describes the PSO-based circuit design, while Section 6.3 exhibits the simulation results.

6.1. Particle Swarm Optimization

In literature about PSO the term ‘swarm intelligence’ appears rather often and, therefore, we begin by explaining why this is so.



(a)

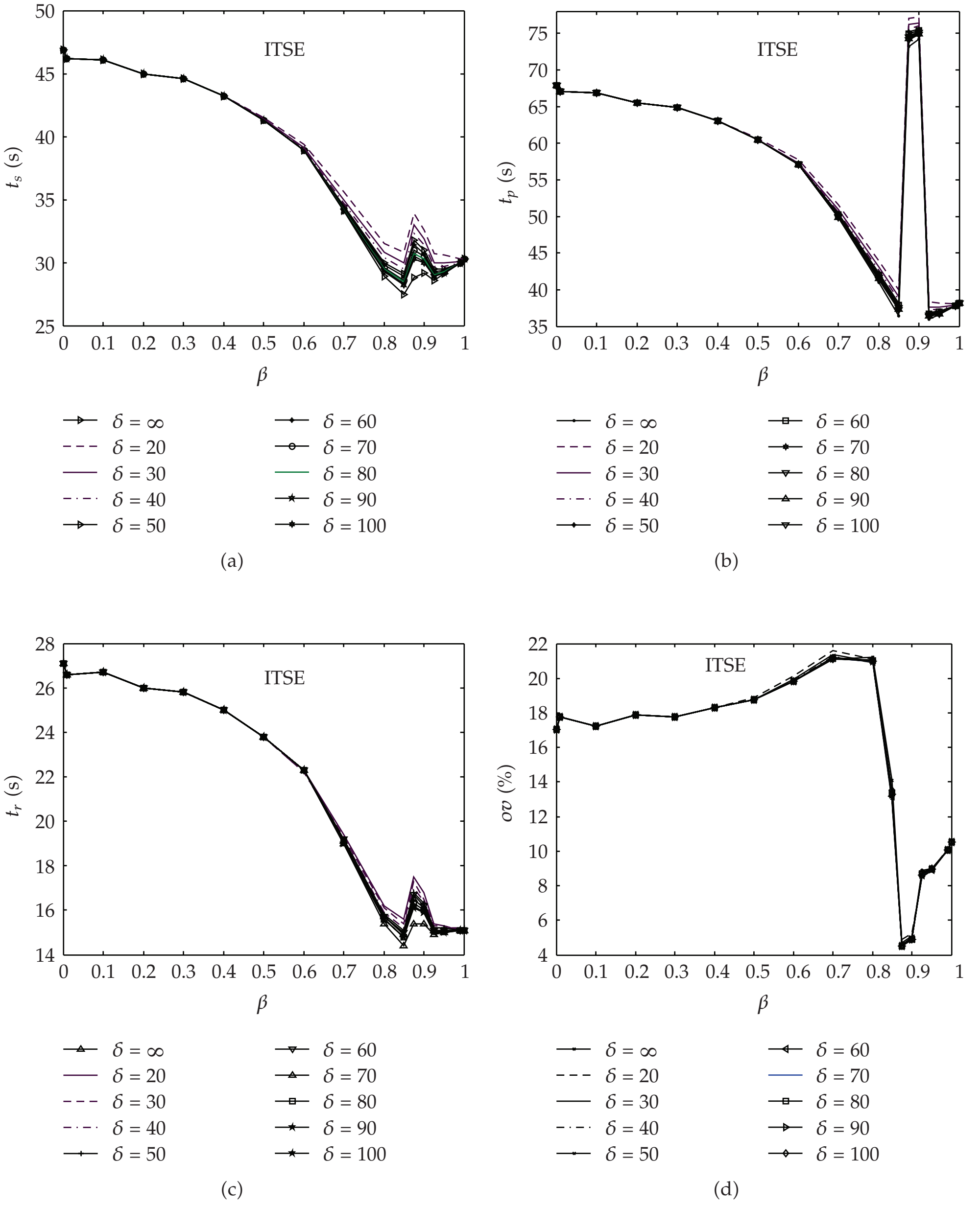
(b)

(c)

(d)

Figure 23: Parameters t_s, t_p, t_r, $ov(\%)$ for the step responses of the closed-loop system for the ISE indice, with a PID$^\beta$ controller, when $\delta = \{20, \ldots, 100\}$ and $\delta = \infty$, $x = 3.0\,\text{m}$.

Noncomputer scientists (ornithologists, biologists, and psychologists) did early research, which led into the theory of particle swarms. In these areas, the term “swarm intelligence” is well known and characterizes the case when a large number of individuals are able of accomplish complex tasks. Motivated by these facts, some basic simulations of swarms were abstracted into the mathematical field. The usage of swarms for solving simple tasks in nature became an intriguing idea in algorithmic and function optimization.



(a)

(b)

(c)

(d)

Figure 24: Parameters t_s, t_p, t_r, $ov(\%)$ for the step responses of the closed-loop system for the ITSE indice, with a PID$^\beta$ controller, when $\delta = \{20, \ldots, 100\}$ and $\delta = \infty$, $x = 3.0\,\mathrm{m}$.

Eberhart and Kennedy were the first to introduce the PSO algorithm [35], which is an optimization method inspired in the collective intelligence of swarms of biological populations, and was discovered through simplified social model simulation of bird flocking, fishing schooling, and swarm theory.

In the PSO, instead of using genetic operators, as in the case of GAs, each particle (individual) adjusts its flying according with its own and its companions experiences. Each particle is treated as a point in a D-dimensional space and is manipulated as described in what follows in the original PSO algorithm:

$$v_{id} = v_{id} + c_1 \text{rand}(\,)(p_{id} - x_{id}) + c_2 \text{Rand}(\,)(p_{gd} - x_{id}), \tag{6.1a}$$

$$x_{id} = x_{id} + v_{id}, \tag{6.1b}$$

where c_1 and c_2 are positive constants, rand( ) and Rand( ) are two random functions in the range $[0,1]$, $X_i = (x_{i1}, x_{i2}, \ldots, x_{i\mathrm{D}})$ represents the ith particle, $P_i = (p_{i1}, p_{i2}, \ldots, p_{i\mathrm{D}})$ is the best previous position (the position giving the best fitness value) of the particle, the symbol g represents the index of the best particle among all particles in the population, and $V_i = (v_{i1}, v_{i2}, \ldots, v_{i\mathrm{D}})$ is the rate of the position change (velocity) for particle i.

However, (6.1a) and (6.1b) represent the flying trajectory of a population of particles. Also, (6.1a) describes how the velocity is dynamically updated and (6.1b) the position update of the “flying” particles. Moreover, (6.1b) is divided in three parts, namely the momentum, the cognitive and the social parts. In the first part the velocity cannot be changed abruptly: it is adjusted based on the current velocity. The second part represents the learning from its own flying experience. The third part consists on the learning group flying experience [36].

The first new parameter added into the original PSO algorithm is the inertia weigh. The dynamic equation of PSO with inertia weigh is modified to be

$$v_{id} = w v_{id} + c_1 \text{rand}(\,)(p_{id} - x_{id}) + c_2 \text{Rand}(\,)(p_{gd} - x_{id}), \tag{6.2a}$$

$$x_{id} = x_{id} + v_{id}, \tag{6.2b}$$

where w constitutes the inertia weigh that introduces a balance between the global and the local search abilities. A large inertia weigh facilitates a global search while a small inertia weigh facilitates a local search.

Another parameter, called constriction coefficient k, is introduced with the hope that it can insure a PSO to converge. A simplified method of incorporating it appears in (6.3), where k is function of c_1 and c_2 as it is presented as follows:

$$\begin{gathered} v_{id} = k\left[v_{id} + c_1 \text{rand}(\,)(p_{id} - x_{id}) + c_2 \text{Rand}(\,)(p_{gd} - x_{id})\right], \\ x_{id} = x_{id} + v_{id}, \end{gathered} \tag{6.3}$$

$$k = 2\left(2 - \phi - \sqrt{\phi^2 - 4\phi}\right)^{-1}, \tag{6.4}$$

where $\phi = c_1 + c_2$, $\phi > 4$.

There are two different PSO topologies, namely, the global version and the local version. In the global version of PSO, each particle flies through the search space with a velocity that is dynamically adjusted according to the particle’s personal best performance achieved so far and the best performance achieved so far by all particles. On the other hand, in the local version of PSO, each particle’s velocity is adjusted according to its personal best and the best performance achieved so far within its neighborhood. The neighborhood of each particle is generally defined as topologically nearest particles to the particle at each side.

```
1. Initialize the population
2. Calculate the fitness of each individual in the population
3. Reproduce selected individuals to form a new population
4. Perform evolutionary operations such as crossover and mutation on the population
5. Loop to step 2 until some condition is met
```

Figure 25: Evolutionary computation algorithm.

```
1. Initialize the population
2. Calculate the fitness of each individual in the population
3. Reproduce selected individuals to form a new population
4. Perform evolutionary operations such as crossover and mutation on the population
5. Apply a local search algorithm
5. Loop to step 2 until some condition is met
```

Figure 26: Memetic algorithm.

PSO is an evolutionary algorithm simple in concept, easy to implement and computationally efficient. Figures 25, 26, and 27 present a generic EC algorithm, a hybrid algorithm, more precisely a MA and the original procedure for implementing the PSO algorithm, respectively.

The different versions of the PSO algorithms are the real-value PSO, which is the original version of PSO and is well suited for solving real-value problems; the binary version of PSO, which is designed to solve binary problems; and the discrete version of PSO, which is good for solving the event-based problems. To extend the real-value version of PSO to binary/discrete space, the most critical part is to understand the meaning of concepts such as trajectory and velocity in the binary/discrete space.

Kennedy and Eberhart [35] use velocity as a probability to determine whether x_{id} (a bit) will be in one state or another (zero or one). The particle swarm formula of (6.1a) remains unchanged, except that now p_{id} and x_{id} are integers in $[0.0, 1.0]$ and a logistic transformation $S(v_{id})$ is used to accomplish this modification. The resulting change in position is defined by the following rule:

$$\text{if } [\text{rand}(\,) < S(v_{id})] \text{ then } x_{id} = 1; \text{ else } x_{id} = 0, \tag{6.5}$$

where the function $S(v)$ is a sigmoid limiting transformation and $\text{rand}(\,)$ is a random number selected from a uniform distribution in the range $[0.0, 1.0]$.

1. Initialize population in hyperspace
2. Evaluate fitness of individual particles
3. Modify velocities based on previous best and global (or neighborhood) best
4. Terminate on some condition
5. Go to step 2

Figure 27: Particle swarm optimization process.

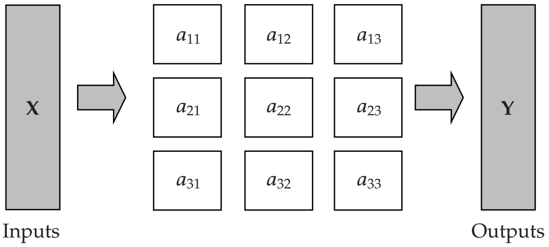


Figure 28: A 3×3 matrix representing a circuit with input X and output Y.

6.2. PSO Based Circuit Design

We adopt a PSO algorithm to design combinational logic circuits. A truth table specifies the circuits and the goal is to implement a functional circuit with the least possible complexity. Four sets of logic gates have been defined, as shown in Table 2, being *Gset 2* the simplest one (i.e., a RISC-like set) and *Gset 6* the most complex gate set (i.e., a CISC-like set). Logic gate named WIRE means a logical no-operation.

In the PSO scheme the circuits are encoded as a rectangular matrix $\mathbf{A}$ ($row \times column = r \times c$) of logic cells as represented in Figure 28.

Three genes represent each cell: *<input1><input2><gate type>*, where *input1* and *input2* are one of the circuit inputs, if they are in the first column, or one of the previous outputs, if they are in other columns. The gate type is one of the elements adopted in the gate set. The chromosome is formed with as many triplets as the matrix size demands (e.g., triplets = $3 \times r \times c$). For example, the chromosome that represents a 3×3 matrix is depicted in Figure 29.

The initial population of circuits (particles) has a random generation. The initial velocity of each particle is initialized with zero. The following velocities are calculated applying (6.2a) and the new positions result from using (6.2b). This way, each potential solution, called particle, flies through the problem space. For each gene is calculated the corresponding velocity. Therefore, the new positions are as many as the number of genes in the chromosome. If the new values of the input genes result out of range, then a re-insertion function is used. If the calculated gate gene is not allowed a new valid one is generated at random. These particles then have memory and each keeps information of its previous best position (*pbest*) and its corresponding fitness. The swarm has the *pbest* of all the particles and the particle with the greatest fitness is called the global best (*gbest*).

The basic concept of the PSO technique lies in accelerating each particle towards its *pbest* and *gbest* locations with a random weighted acceleration. However, in our case we also use a kind of mutation operator that introduces a new cell in 10% of the population. This mutation operator changes the characteristics of a given cell in the matrix. Therefore, the

0	1	2		24	25	26	Genes
Input	Input	Gate	…	Input	Input	Gate	
a_{11}			…	a_{33}			Matrix element

Figure 29: Chromosome for the 3 × 3 matrix of Figure 28.

mutation modifies the gate type and the two inputs, meaning that a completely new cell can appear in the chromosome.

To run the PSO we have also to define the number P of individuals to create the initial population of particles. This population is always the same size across the generations, until reaching the solution.

The calculation of the fitness function F_s in (6.6) has two parts, f_1 and f_2, where f_1 measures the functionality and f_2 measures the simplicity. In a first phase, we compare the output Y produced by the PSO-generated circuit with the required values Y_R, according with the truth table, on a bit-per-bit basis. By other words, f_1 is incremented by one for each correct bit of the output until f_1 reaches the maximum value f_{10} that occurs when we have a functional circuit. Once the circuit is functional, in a second phase, the algorithm tries to generate circuits with the least number of gates. This means that the resulting circuit must have as much genes <*gate type*> ≡<*wire*> as possible. Therefore, the index f_2, that measures the simplicity (the number of null operations), is increased by *one* (*zero*) for each *wire* (*gate*) of the generated circuit, yielding

$$\begin{gathered}
f_{10} = 2^{ni} \times no, \\
f_1 = f_1 + 1 \text{ if } \{\text{bit } i \text{ of } \mathbf{Y}\} = \{\text{bit } i \text{ of } \mathbf{Y_R}\}, \quad i = 1, \dots, f_{10}, \\
f_2 = f_2 + 1 \text{ if } gate\ type = wire, \\
F_s = \begin{cases} f_1, & F_s < f_{10}, \\ f_1 + f_2, & F_s \geq f_{10}, \end{cases}
\end{gathered} \tag{6.6}$$

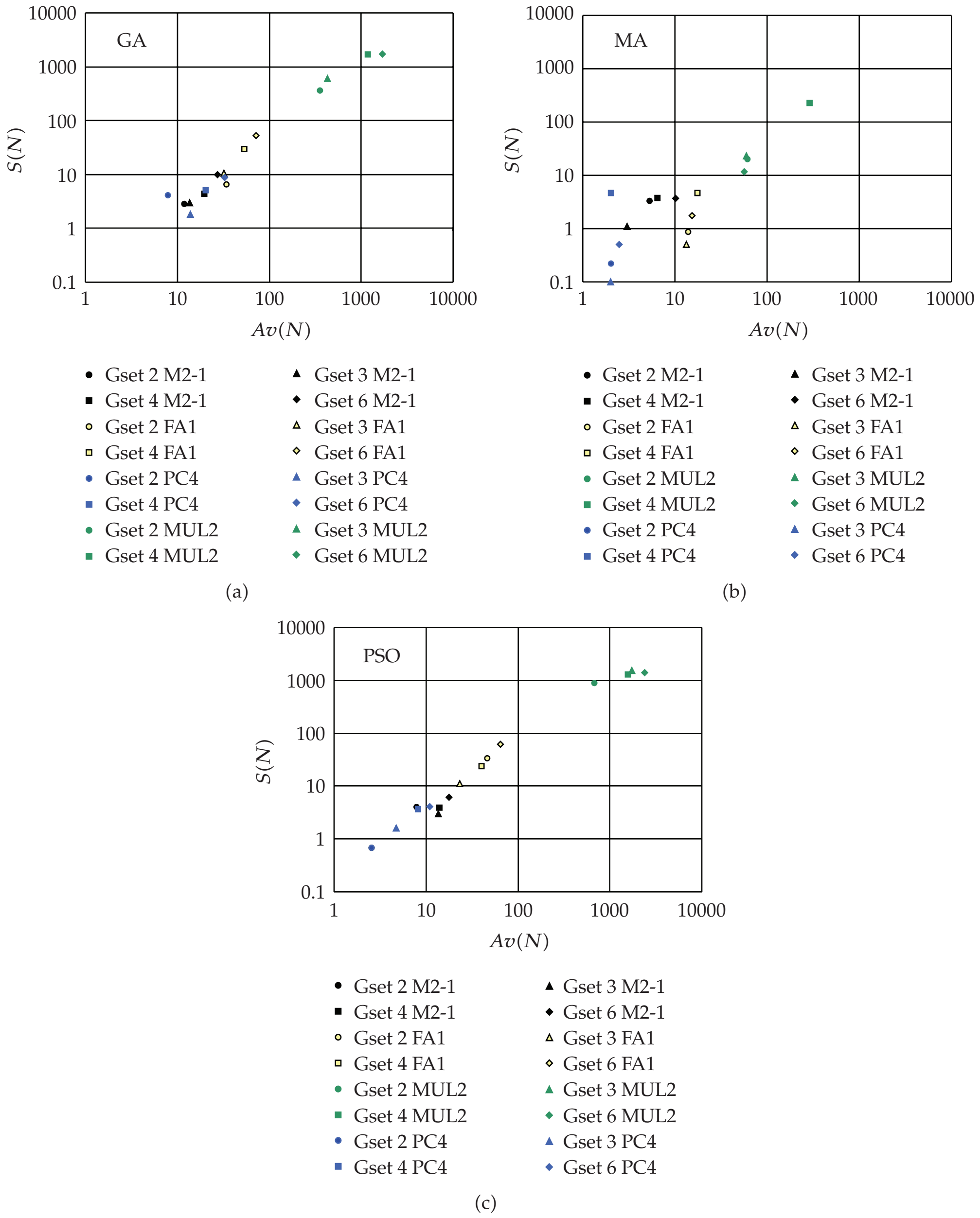
where *ni* and *no* represent the number of inputs and outputs of the circuit.

The concept of dynamic fitness function F_d results from an analogy between control systems and the GA case, where we master the population through the fitness function. The simplest control system is the proportional algorithm; nevertheless, there can be other control algorithms, such as the proportional and the differential scheme.

In this line of thought, (6.6) is a static fitness function F_s and corresponds to using a simple proportional algorithm. Therefore, to implement a proportional-derivative evolution the fitness function needs a scheme of the type [18]

$$F_d = F_s + KD^{\mu}[F_s], \tag{6.7}$$

where $0 \leq \mu \leq 1$ is the differential fractional-order and $K \in \mathfrak{R}$ is the "gain" of the dynamical term.



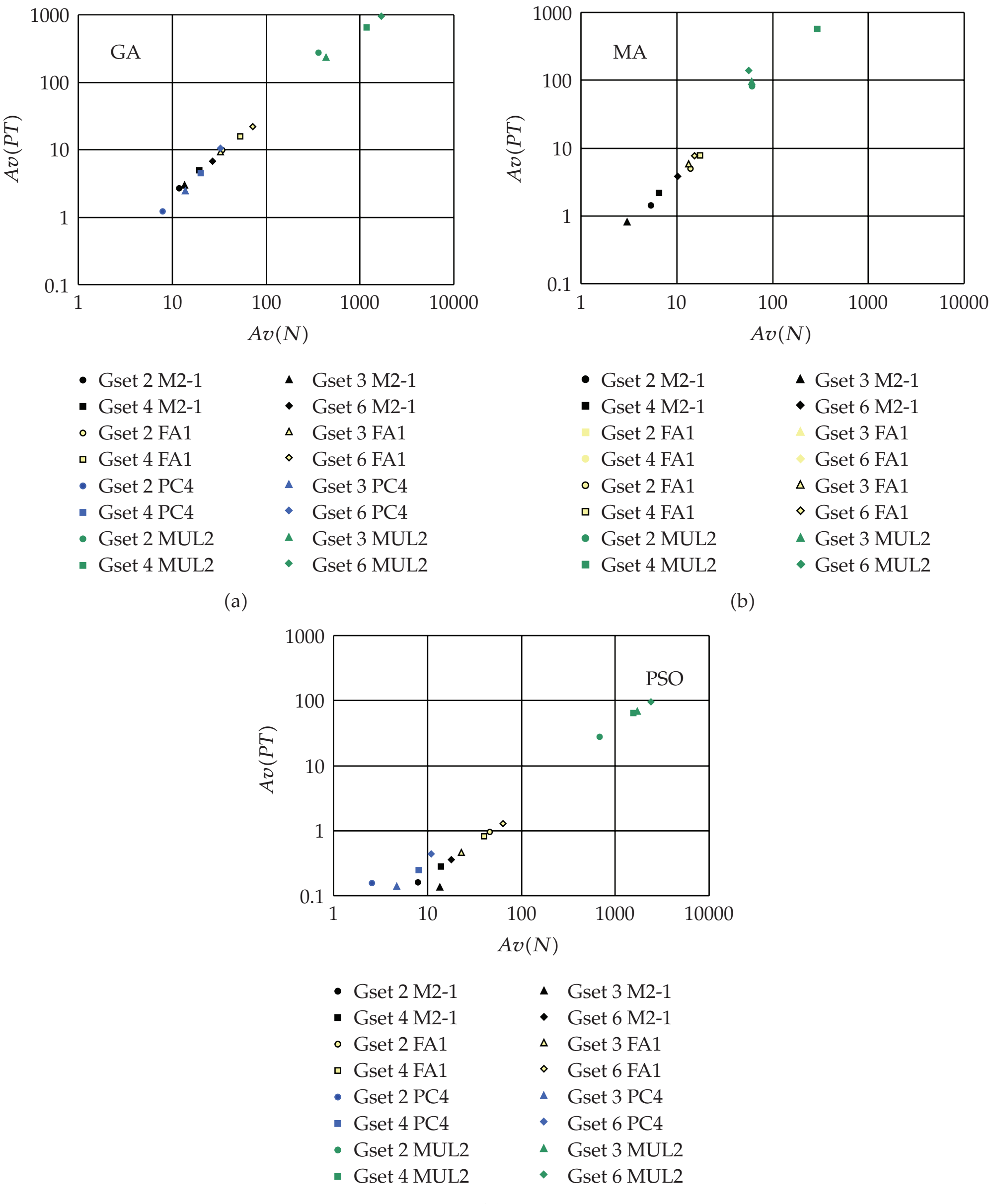
(a) (b)

(c)

Figure 30: $S(N)$ versus $Av(N)$ with $P = 3000$ and F_s for the GA, the MA, and the PSO algorithms.

6.3. Experiments and Results

A reliable execution and analysis of an EC algorithm usually requires a large number of simulations to provide a reasonable assurance that the stochastic effects are properly considered. Therefore, in this study are developed $n = 20$ simulations for each case under analysis.



(a)

(b)

(c)

Figure 31: $Av(PT)$ versus $Av(N)$ with $P = 3000$ and F_s for the GA, the MA, and the PSO algorithms.

The experiments consist on running the three algorithms $\{\mathrm{GA}, \mathrm{MA}, \mathrm{PSO}\}$ to generate a typical combinational logic circuit, namely, a 2-to-1 multiplexer ($M2$-1), a 1-bit full adder (*FA1*), a 4-bit parity checker (*PC4*) and a 2-bit multiplier (*MUL2*), using the fitness scheme described in (6.6) and (6.7). The circuits are generated with the gate sets presented in Table 2 and $P = 3000$, $w = 0.5$, $c_1 = 1.5$, and $c_2 = 2$.

Figure 30 depicts the standard deviation of the number of generations to achieve the solution $S(N)$ versus the average number of generations to achieve the solution $Av(N)$

Table 2: Gate sets.

Gate set	Logic gates
Gset 2	{AND, XOR, WIRE}
Gset 3	{AND, OR, XOR, WIRE}
Gset 4	{AND, OR, XOR, NOT, WIRE}
Gset 6	{AND, OR, XOR, NOT, NAND, NOR, WIRE}

Table 3: The parameters (a, b) and (c, d).

Algorithm	a	b	c	d
GA	0.0365	1.602	0.1526	1.1734
MA	0.0728	1.2602	0.2089	1.3587
PSO	0.2677	1.1528	0.0141	1.1233

for the algorithms $\{\text{GA}, \text{MA}, \text{PSO}\}$, the circuits $\{M2\text{-}1, FA1, PC4, MUL2\}$, and the gate sets $\{2, 3, 4, 6\}$. In these figure, we can see that the *MUL2* circuit is the most complex one, while the *PC4* and the $M2$-1 are the simplest circuits. It is also possible to conclude that *Gset 6* is the less efficient gate set for all algorithms and circuits.

Figure 30 reveals that the plots follow a power law:

$$S(N) = a[Av(N)]^b \quad a, b \in \Re. \tag{6.8}$$

Table 3 presents the numerical values of the parameters (a, b) for the three algorithms.

In terms of $S(N)$ versus $Av(N)$, the MA algorithm presents the best results for all circuits and gate sets. In what concerns the other two algorithms, the PSO is superior (inferior) to the GA for complex (simple) circuits.

Figure 31 depicts the average processing time to obtain the solution $Av(PT)$ versus the average number of generations to achieve the solution $Av(N)$ for the algorithms $\{\text{GA}, \text{MA}, \text{PSO}\}$, the circuits $\{M2\text{-}1, FA1, PC4, MUL2\}$ and the gate sets $\{2, 3, 4, 6\}$. When analysing these charts it is clear that the PSO algorithm demonstrates to be around ten times faster than the MA and the GA algorithms.

These plots follow also a power law:

$$Av(PT) = c[Av(N)]^d \quad c, d \in \Re. \tag{6.9}$$

Table 3 shows parameters (c, d) and we can see that the PSO algorithm has the best values.

Figures 32 and 33 depict the standard deviation of the number of generations to achieve the solution $S(N)$ and the average processing time to obtain the solution $Av(PT)$, respectively, versus the average number of generations to achieve the solution $Av(N)$ for the PSO algorithm using F_d, the circuits {*M2-1, FA1, PC4, MUL2*}, and the gate sets $\{2, 3, 4, 6\}$. We conclude that F_d leads to better results in particular for the *MUL2* circuit and for the $Av(PT)$.

Figures 34 and 35 present a comparison between F_s and F_d.

In terms of $S(N)$ versus $Av(N)$ it is possible to say that the MA algorithm presents the best results. Nevertheless, when analysing Figure 31, that shows $Av(PT)$ versus $Av(N)$ for reaching the solutions, we verify that the PSO algorithm is very efficient, in particular, for the more complex circuits.

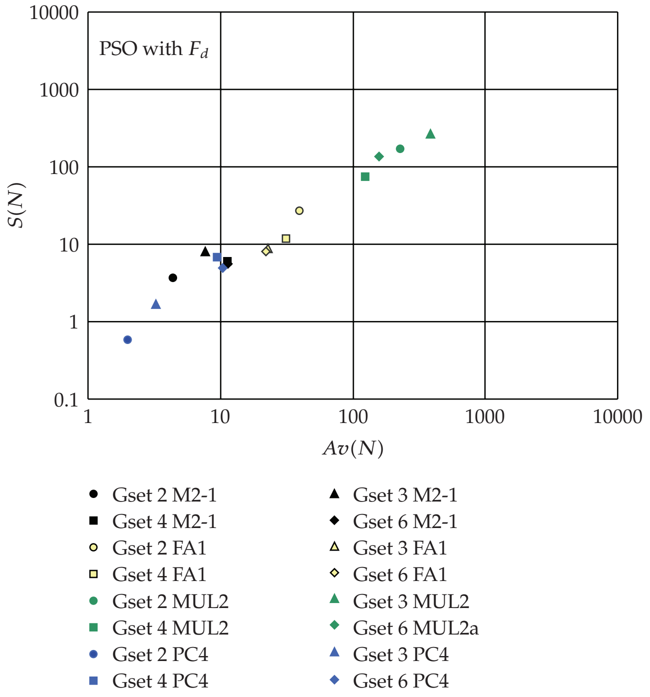


Figure 32: $S(N)$ versus $Av(N)$ for the PSO algorithm, $P = 3000$ and F_d.

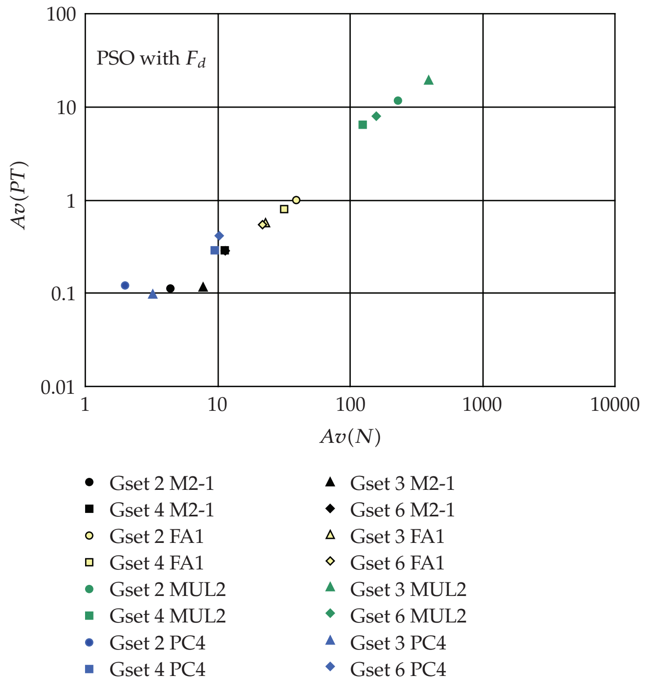


Figure 33: $Av(PT)$ versus $Av(N)$ for the GA, $P = 3000$ and F_d.

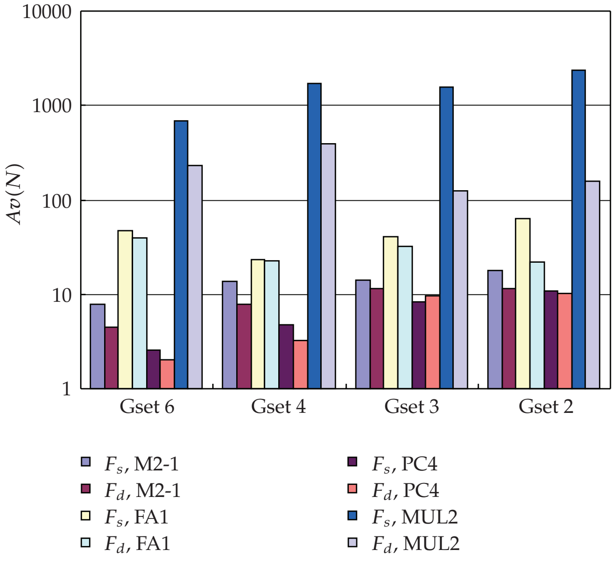


Figure 34: $Av(N)$ for the PSO algorithm, $P = 3000$ using F_s and F_d.

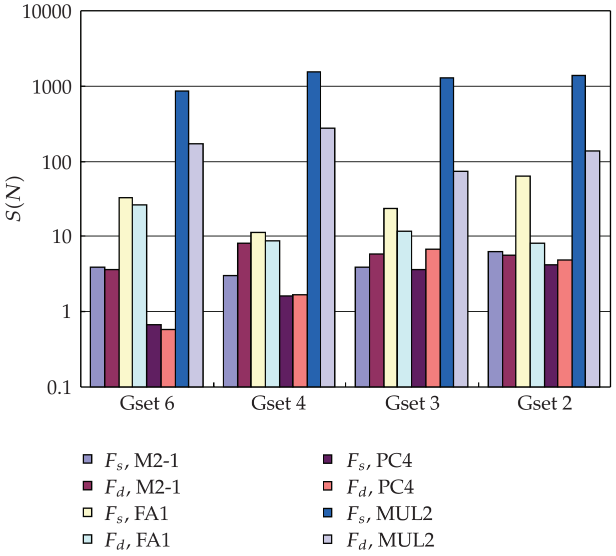


Figure 35: $S(N)$ for the PSO algorithm, $P = 3000$ using F_s and F_d.

The PSO-based algorithm for the design of combinational circuits follows the same profile as the other two evolutionary techniques presented in this paper.

Adopting the study of the $S(N)$ versus $Av(N)$ for the three evolutionary algorithms, the MA algorithm presents better results over the GA and the PSO algorithms. However, in what concerns the processing time to achieve the solutions, the PSO outcomes clearly the GA and the MA algorithms. Moreover, applying the F_d the results obtained are improved further in all gate sets and in particular for the more complex circuits.

7. Conclusions

Fractional Calculus (FC) goes back to the beginning of the theory of differential calculus. Nevertheless, the application of FC just emerged in the last two decades, due to the progress in the area of chaos that revealed subtle relationships with the FC concepts.

Recently FC has been a fruitful field of research in science and engineering and many scientific areas are currently paying wider attention to the FC concepts. In the field of dynamical systems theory, some work has been carried out but the proposed models and algorithms are still in a preliminary stage of establishment. This article presented several case studies on the implementation of FC-based models and control systems, being demonstrated the advantages of using the FC theory in different areas of science and engineering. In fact, this paper studied a variety of different physical systems, namely

(i) tuning of PID controllers using fractional calculus concepts;

(ii) fractional PD$^{\alpha}$ control of a hexapod robot;

(iii) fractional dynamics in the trajectory control of redundant manipulators;

(iv) heat diffusion;

(v) circuit synthesis using evolutionary algorithms.

It has been recognized the advantageous use of this mathematical tool in the modeling and control of these dynamical systems, and the results demonstrate the importance of Fractional Calculus and motivate for the development of new applications.

References

[1] K. B. Oldham and J. Spanier, *The Fractional Calculus: Theory and Applications of Differentiation and Integration to Arbitrary Order*, vol. 11 of *Mathematics in Science and Engineering*, Academic Press, New York, NY, USA, 1974.

[2] K. S. Miller and B. Ross, *An Introduction to the Fractional Calculus and Fractional Differential Equations*, A Wiley-Interscience Publication, John Wiley & Sons, New York, NY, USA, 1993.

[3] I. Podlubny, *Fractional Differential Equations*, vol. 198 of *Mathematics in Science and Engineering*, Academic Press, San Diego, Calif, USA, 1999.

[4] R. Hilfer, Ed., *Applications of Fractional Calculus in Physics*, World Scientific Publishing, Singapore, 2000.

[5] A. Oustaloup, *La Commande CRONE: Commande Robuste d'Ordre Non Entier*, Editions Hermès, Paris, France, 1991.

[6] A. Oustaloup, *La Dérivation Non Entière: Théorie, Synthèse et Applications*, Editions Hermès, Paris, France, 1995.

[7] R. S. Barbosa, J. A. T. Machado, and I. M. Ferreira, "PID controller tuning using fractional calculus concepts," *Fractional Calculus & Applied Analysis*, vol. 7, no. 2, pp. 119–134, 2004.

[8] R. S. Barbosa, J. A. T. Machado, and I. M. Ferreira, "Tuning of PID controllers based on bode's ideal transfer function," *Nonlinear Dynamics*, vol. 38, no. 1–4, pp. 305–321, 2004.

[9] M. F. Silva, J. A. T. Machado, and A. M. Lopes, "Comparison of fractional and integer order control of an hexapod robot," in *Proceedings of International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 5 of *19th Biennial Conference on Mechanical Vibration and Noise*, pp. 667–676, ASME, Chicago, Ill, USA, September 2003.

[10] M. F. Silva, J. A. T. Machado, and I. S. Jesus, "Modelling and simulation of walking robots with 3 dof legs," in *Proceedings of the 25th IASTED International Conference on Modelling, Identification and Control (MIC '06)*, pp. 271–276, Lanzarote, Spain, 2006.

[11] M. F. Silva, J. A. T. Machado, and A. M. Lopes, "Position/force control of a walking robot," *Machine Intelligence and Robot Control*, vol. 5, pp. 33–44, 2003.

[12] M. F. Silva and J. A. T. Machado, "Fractional order PD$^{\alpha}$ joint control of legged robots," *Journal of Vibration and Control*, vol. 12, no. 12, pp. 1483–1501, 2006.

[13] F. Duarte and J. A. T. Machado, “Chaotic phenomena and fractional-order dynamics in the trajectory control of redundant manipulators,” *Nonlinear Dynamics*, vol. 29, no. 1–4, pp. 315–342, 2002.

[14] J. A. T. Machado, “Analysis and design of fractional-order digital control systems,” *Systems Analysis Modelling Simulation*, vol. 27, no. 2-3, pp. 107–122, 1997.

[15] J. A. T. Machado, “Discrete-time fractional-order controllers,” *Fractional Calculus & Applied Analysis*, vol. 4, no. 1, pp. 47–66, 2001.

[16] J. A. T. Machado, I. S. Jesus, J. B. Cunha, and J. K. Tar, “Fractional dynamics and control of distributed parameter systems,” *Intelligent Systems at the Service of Mankind*, vol. 2, pp. 295–305, 2006.

[17] I. S. Jesus, R. S. Barbosa, J. A. T. Machado, and J. B. Cunha, “Strategies for the control of heat diffusion systems based on fractional calculus,” in *Proceedings of the IEEE International Conference on Computational Cybernetics (ICCC ’06)*, Budapest, Hungary, 2006.

[18] C. Reis, J. A. T. Machado, and J. B. Cunha, “Evolutionary design of combinational circuits using fractional-order fitness,” in *Proceedings of the 5th Nonlinear Dynamics Conference (EUROMECH ’05)*, pp. 1312–1321, 2005.

[19] H. W. Bode, *Network Analysis and Feedback Amplifier Design*, Van Nostrand, New York, NY, USA, 1945.

[20] E. S. Conkur and R. Buckingham, “Clarifying the definition of redundancy as used in robotics,” *Robotica*, vol. 15, no. 5, pp. 583–586, 1997.

[21] S. Chiaverini, “Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators,” *IEEE Transactions on Robotics and Automation*, vol. 13, no. 3, pp. 398–410, 1997.

[22] C. A. Klein and C. C. Huang, “Review of pseudoinverse control for use with kinematically redundant manipulators,” *IEEE Transactions on Systems, Man and Cybernetics*, vol. 13, no. 2, pp. 245–250, 1983.

[23] T. Yoshikawa, *Foundations of Robotics: Analysis and Control*, MIT Press, Cambridge, Mass, USA, 1988.

[24] J. S. Bay, “Geometry and prediction of drift-free trajectories for redundant machines under pseudoinverse control,” *International Journal of Robotics Research*, vol. 11, no. 1, pp. 41–52, 1992.

[25] R. G. Roberts and A. A. Maciejewski, “Singularities, stable surfaces, and the repeatable behavior of kinematically redundant manipulators,” *International Journal of Robotics Research*, vol. 13, no. 1, pp. 70–81, 1994.

[26] K. L. Doty, C. Melchiorri, and C. Bonivento, “A theory of generalized inverses applied to robotics,” *International Journal of Robotics Research*, vol. 12, no. 1, pp. 1–19, 1993.

[27] Y. Nakamura, *Advanced Robotics: Redundancy and Optimization*, Addinson-Wesley, New York, NY, USA, 1991.

[28] B. Siciliano, “Kinematic control of redundant robot manipulators: a tutorial,” *Journal of Intelligent and Robotic Systems*, vol. 3, no. 3, pp. 201–212, 1990.

[29] W. J. Chung, Y. Yorm, and W. K. Chung, “Inverse kinematics of planar redundant manipulators via virtual links with configuration index,” *Journal of Robotic Systems*, vol. 11, no. 2, pp. 117–128, 1994.

[30] S. Seereeram and J. T. Wen, “A global approach to path planning for redundant manipulators,” *IEEE Transactions on Robotics and Automation*, vol. 11, no. 1, pp. 152–159, 1995.

[31] M. da Graça Marcos, F. B. M. Duarte, and J. A. T. Machado, “Complex dynamics in the trajectory control of redundant manipulators,” *Nonlinear Science and Complexity*, pp. 134–143, 2007.

[32] S. J. Louis, G. J. E. Rawlins, and G. J. Designer, “Genetic algorithms: genetic algorithms in structure design,” in *Proceedings of the 4th International Conference on Genetic Algorithms*, 1991.

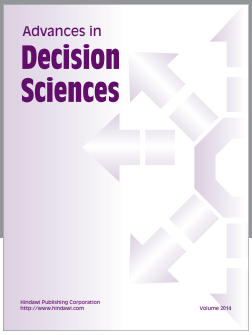
[33] D. E. Goldberg, *Genetic Algorithms in Search Optimization and Machine Learning*, Addison-Wesley, New York, NY, USA, 1989.

[34] C. Reis, J. A. T. Machado, and J. B. Cunha, “An evolutionary hybrid approach in the design of combinational digital circuits,” *WSEAS Transactions on Systems*, vol. 4, no. 12, pp. 2338–2345, 2005.

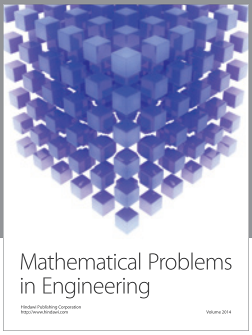
[35] J. Kennedy and R. C. Eberhart, “Particle swarm optimization,” in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, November 1995.

[36] Y. Shi and R. C. Eberhart, “A modified particle swarm optimizer,” in *Proceedings of the International Conference on Evolutionary Computation*, pp. 69–73, May 1998.

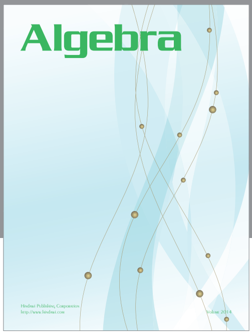
Advances in Operations Research



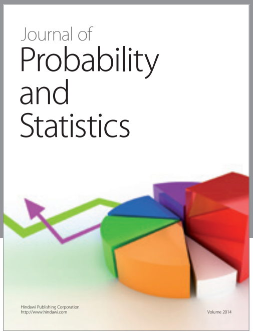
Advances in Decision Sciences



Mathematical Problems in Engineering



Algebra

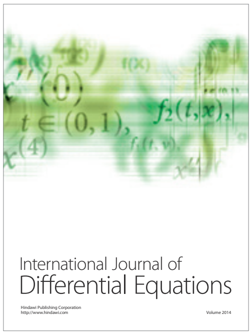


Journal of Probability and Statistics

The Scientific World Journal

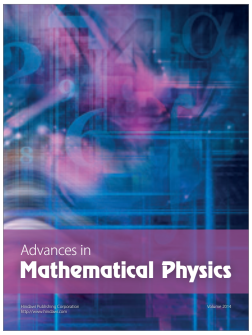
Hindawi

Submit your manuscripts at
http://www.hindawi.com

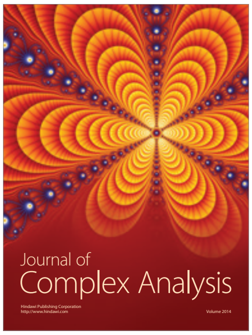


International Journal of Differential Equations

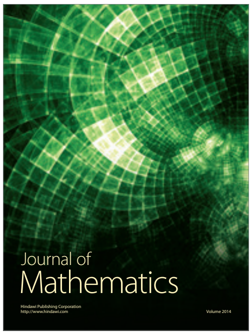
International Journal of Combinatorics



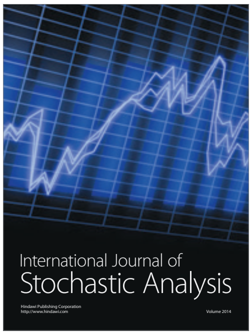
Advances in Mathematical Physics



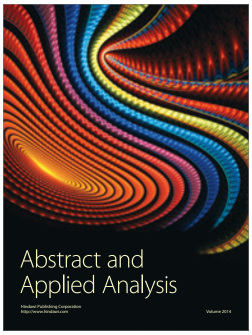
Journal of Complex Analysis



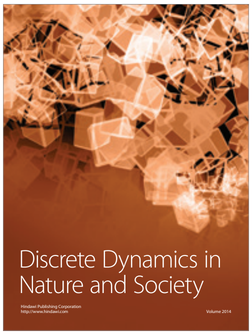
Journal of Mathematics



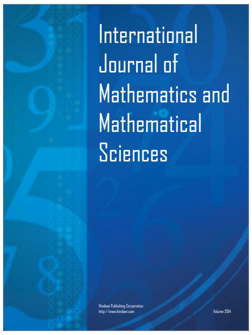
International Journal of Stochastic Analysis



Abstract and Applied Analysis



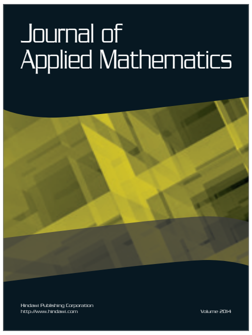
Discrete Dynamics in Nature and Society



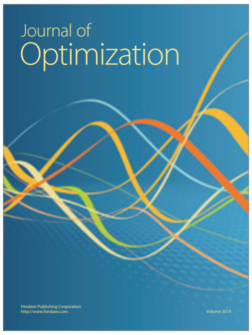
International Journal of Mathematics and Mathematical Sciences

Journal of Discrete Mathematics

Journal of Function Spaces



Journal of Applied Mathematics



Journal of Optimization