DOI 10.2478/ama-2025-0031

HARDWARE IMPLEMENTATION OF FRACTIONAL-ORDER CALCULUS IN CONTROL SYSTEMS

Paweł KIECZMERSKI*

*Department of Automation and Robotics, Białystok University of Technology, Wiejska 45A, 15-351 Białystok, Poland

pawel.kieczmerski@pb.edu.pl

received 10 September 2024, revised 18 April 2025, accepted 2 May 2025

Abstract: This article analyzes the implementation process of a fractional-order control system using available toolboxes, software, and hardware. The main objective is to showcase the current state-of-the-art hardware implementation of fractional-order control, comparing its potential to classical counterparts, and emphasizing the benefits of its utilization in industry. The article covers theoretical aspects of fractionalorder calculus and provides an example implementation of a Fractional-order PID controller with a NI MyRIO-1900 measurement board with FPGA module, comparing it with simulations for a given control plant.

Key words: fractional calculus, fractional-order systems, fractance; domino ladder, fractional integrator, Simulink, implementation, control

1. INTRODUCTION

In recent years, fractional calculus has gained considerable attention among researchers in various fields of science and technology. This calculus deals with the general extension of the concept of differentiation and integration to non-integral orders, which makes it possible to analyze and model complex phenomena that are not fully described by classical equations [1, 2, 3, 4, 5]. However, the greatest potential of fractional calculus lies in its application to systems modeling and control, where the possibility of designing much more robust and accurate control systems emerges. Current research indicates that fractional-order controllers can achieve much better values of control quality indicators and are more resistant to changes in model parameters than the commonly used classical PID controller [2, 6, 7, 8, 9, 10, 11, 12, 13], but most industrial solutions available on the market use PI/PID-type controllers without achieving optimal performance. This fact is due to the difficulty of understanding the complexity of non-integer order controllers by the personnel operating the system and the complexity of implementing control algorithms in classical microcontrollers or other commercial devices. With the coming of Industry 4.0 [14] the automation industry is looking for new solutions to increase overall efficiency; that is why this paper will present both the recent available tools and implementation options for non-integer controllers.

The purpose of this article is to compile key theoretical foundations of fractional-order calculus, provide an overview of current trends in its practical applications, and demonstrate an example implementation of Fractional-order PID controller on myRIO-1900 measurement board.

In this paper, all necessary theoretical aspects related to fractional-order calculus and its application in automatic control systems will be presented. The most popular types of fractional-order controllers, methods for approximating fractional order, and current tuning methods for such controllers will be discussed. This will be followed by an analysis of current trends in hardware implementation of fractional-order controllers, along with an indication and comparison of available hardware and software tools. These tools were used to obtain models, the responses of which were compared with the response of a model of an object with fractional-order dynamics, called fractance, in order to determine the best one. The best tool will be used to develop an automatic control system with both a classical PID controller and a fractional-order controller, both in simulation and in the laboratory, and their results will eventually be compared. Both types of controllers will be tuned using the Modified Grey Wolf optimization algorithm.

2. FRACTIONAL CALCULUS

The fractional calculus is a form of generalization of the operations of integration and differentiation, through a new operator ${}_aD_t^q$ combining the functionality of both, where q - the order of the operator, such that q is a real number q ∈ R and a and t are the limits of the operation. Depending on the sign of the order of the operator, it can be treated as a differentiation or integration operator [2] according to the following formula:

$${}_aD_t^q = \begin{cases} \frac{d^q}{dt^q} & dla\ q > 0, \\ 1 & dla\ q = 0, \\ \int_a^t (d\tau)^q & dla\ q < 0. \end{cases} \tag{2.1}$$

There are many definitions of fractional operator, but they are used in different fields, making it easier to analyze the relevant functions. Currently, the most popular definitions found in modern books describing fractional calculus are the Riemann-Liovile definition RL, the Grünwald-Letnikov definition GL and the Caputo definition C.

Definition 2.1. A function given by the integral [1]

$$\Gamma(x) = \int_0^\infty t^{x-1} e^{-t} dt,\ R(x) > 0 \tag{2.2}$$

is called the Euler gamma function and satisfies the equality

$$\Gamma(x+1) = x\Gamma(x)$$

where $R(x)$ is real part of complex number x. (2.3)

Definition 2.2. Riemann-Liouville fractional-order derivative is defined as [1]

$$ {}^{RL}_{a}D^{q}_{t} f(t) = \frac{1}{\Gamma(N-q)} \frac{dN}{dt^N} \int_a^t (t-\tau)^{N-q-1} f(\tau) d\tau. \quad (2.4)$$

Definition 2.3. Grünwald–Letnikov fractional-order derivative is defined as [4]

$$ {}^{GL}_{a}D^{q}_{t} f(t) = \lim_{h \to 0} h^{-q} \sum_{k=0}^{\left[\frac{t-a}{h}\right]} (-1)^k \binom{q}{k} f(t-kh). \quad (2.5)$$

Definition 2.4. Caputo fractional-order derivative is defined as [1]

$$ {}^{C}_{a}D^{q}_{t} f(t) = \frac{1}{\Gamma(N-q)} \int_a^t \frac{f^{(N)}(\tau)}{(t-\tau)^{q+1-N}} d\tau, (n-1 \leq q < n). \quad (2.6)$$

Due to its form, the Riemann-Liouville definition is used for relatively simple functions (xa, ex, sin(x), etc.), while Grünwald-Letnikov definition found its use in numerical evaluation. On the other hand, because of its ability to consider time delays, memory effects, and the possibility of using the same form of initial conditions as for the integer-order case, the Caputo definition has found wide application in control theory and automation. In this paper, the Caputo definition will be used precisely because of the aforementioned properties.

Definition 2.5. The formula for the Laplace transform of the q−th order Caputo derivative eq. (2.6), when (N − 1 ⩽ q < N) has the form [1]:

$$ \mathcal{L}\left[{}^{C}_{0}D^{q}_{t} f(t) \right] = s^q F(s) - \sum_{k=1}^{N} s^{(q-k)} f^{(k-1)}(0^+). \quad (2.7)$$

Given that the function f(t) and all its derivatives have zero initial conditions f(0) = 0 when t = 0, then transform of the q−th order Caputo derivative can be simplified to $s^q F(s)$.

3. FRACTIONAL-ORDER CONTROLLERS

Controllers are an integral part of systems that provide process control using feedback loops. Fractional-order controllers, which are a generalization of classical integer-order controllers, are becoming increasingly popular. The leading types of Fractional-order controllers [4] today are CRONE Controller developed by Oustaloup in 1995, Fractional-order PID controller proposed by Igor Podlubny, and a number of lesser used controllers; Fractional lead-lag compensator (Raynaud and Zegaïnoh, 2000), non-integer integral (Manabe, 1961) and TID compensator (Lurie, 1994). In this paper only FOPID controller will be considered due to its relative simplicity.

3.1. FOPID Controller

Fractional-order PIλDμ controller (FOPID) [3, 4] was first introduced in 1999 by Igor Podlubny as generalization of the classical PID controller with integrator of real order λ and differentiator of real order μ.

Definition 3.1. The FOPID Controller [4, 15] transfer function can be defined in time domain as:

$$ C(s) = k_p + \frac{k_i}{s^\lambda} + k_d s^\mu, \ (\lambda, \mu \geq 0) \quad (3.1)$$

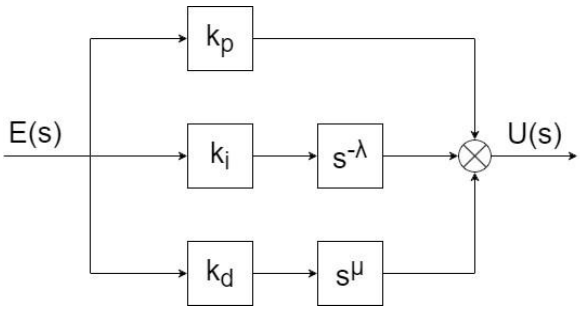

Fig. 3.1. General structure of FOPID Controller

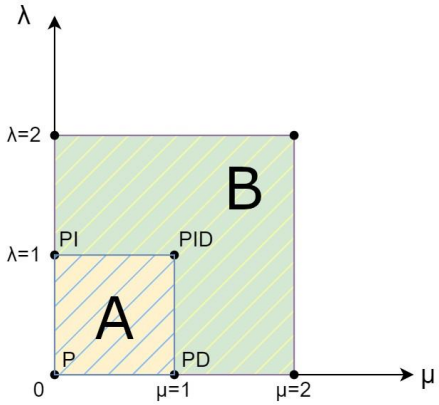

Fig. 3.2. The FOPID controller plane

where, k_p, k_i, k_d are proportional constant, integration constant and differentiation constant respectively.

The general structure of FOPID Controller corresponding to eq. (3.1) is shown on Figure 3.1, where $E(s)$ is control error and $U(s)$ is controller output. In special circumstances $PI^{\lambda}D^{\mu}$ controller can be equal to the classic PID Controller, for example if $\lambda = 1$ and $\mu = 0$, then eq. (3.1) will describe PI controller as shown in Figure 3.2. The controller plane for FOPID has to be divided into two areas: A when $0 < \lambda, \mu \leqslant 1$ and B when $1 < \lambda, \mu \leqslant 2$. This is due to the stability conditions [16] of the system depending on the value of the controller order. In Figure 3.3, where q corresponds to fractional operator order, stability regions are depicted for the controller order corresponding to the zones separated in Figure 3.2. It is worth mentioning that using $\lambda, \mu > 2$ will result in controllers instability.

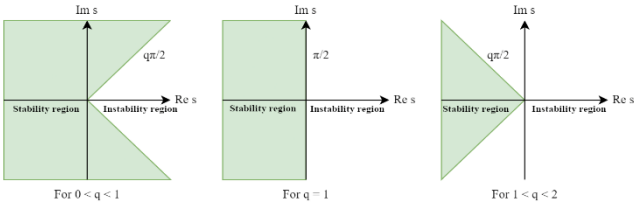

Fig. 3.3. Stability region of fractional order system

3.2. Approximation

For practical implementation of fractional integro-differentiator of order α it is necessary to use conventional transfer function approximation. One of such approximation methods is Oustaloup recursive filter [2,3], which provides very good results in specified frequency range (ω_b; ω_h) using the following equations:

DOI 10.2478/ama-2025-0031

$$s^{\alpha} \approx K \prod_{k=1}^{N} \frac{s+\omega_k'}{s+\omega_k} \tag{3.2}$$

$$\omega_k' = \omega_b \cdot \omega_u^{\frac{2k-1-\alpha}{N}}, \tag{3.3}$$

$$\omega_k = \omega_b \cdot \omega_u^{\frac{2k-1+\alpha}{N}}, \tag{3.4}$$

$$K = \omega_h^{\alpha},\ \omega_u = \sqrt{\omega_h/\omega_b}. \tag{3.5}$$

where, N - order of approximation, ω_h - higher bound of frequency fitness, ω_b - bottom bound of frequency fitness.

For digital implementation it is possible to use any suitable method for conversion to discrete-time equivalent.

3.3. Controller tuning methods

The most challenging part of working with FOPID controllers arises when tuning them with the addition of two parameters related to the orders of integration and differentiation. The authors of the book [3] state that the known principles of the Ziegler-Nichols tuning method for the classical PID, can be applied to FOPID controllers for the values of constants k_p, k_i, k_d, while the orders of λ and μ must be selected experimentally. However, this creates a difficulty in obtaining optimal system performance, as well as requiring the selection of the appropriate type of controller before the tuning process begins, as each controller (P, PI^{λ}, PD^{μ}, $PI^{\lambda}D$) provided significantly different results. They also proposed an analytical tuning method which is as follows:

Consider the mathematical model of a given plant

$$G(s) = \frac{K}{Ts+1}. \tag{3.6}$$

Since open-loop transfer function L(s) = C(s)G(s), the controller transfer function C(s), that gives the Bode's ideal loop transfer function

$$L(s) = \left(\frac{\omega_c}{s}\right)^{\alpha}, \tag{3.7}$$

takes the form:

$$C(s) = K_p \frac{Ts+1}{s^{\alpha}}, \tag{3.8}$$

which can be expressed as

$$C(s) = K_p \left(\frac{1}{s^{\alpha}} + Ts^{1-\alpha}\right). \tag{3.9}$$

The transfer function eq. (3.9) is basically a IαDμ controller (where μ = 1−α). The time-domain equation of the controller C(s) is:

$$u(t) = K_p({}_0D_t^{-\alpha} e(t) + T_0 D_t^{1-\alpha} e(t)). \tag{3.10}$$

The open-loop transfer function L(s) = C(s)G(s) can then be described as

$$L(s) = \frac{K_p K}{s^{\alpha}},\ 1 < \alpha < 2. \tag{3.11}$$

To obtain α and Kp the following procedure has to be used:

1. Find the fractional order α by using formula:

$$\varphi_m = \pi + arg L(j\omega_c) = \pi\left(1 - \frac{\alpha}{2}\right) \tag{3.12}$$

from the desired phase margin φ_m.

2. Calculate the proportional gain K_p by using formula:

$$|L(j\omega_c)| = 1 \rightarrow K_p = \frac{{\omega_c}^{\alpha}}{K}, \tag{3.13}$$

from the gain-crossover frequency ω_{cg} and nominal gain of the system K. This method was verified by experiment conducted by the authors which proved that the system behaved according to designed specifications.

In [17] the extension of Z-N tuning rules was presented, in which the authors pointed out new more complex rules. They stated that by formulating five design criteria—target gain crossover frequency ω_{cg}, desired phase margin φ_m, high-frequency noise attenuation, disturbance rejection, and robustness to plant gain variations—the parameters of a FOPID controller can be systematically computed.

The first criterion is posed as the objective function, while the remaining four are treated as constraints, resulting in a constrained optimization problem. This solution can provide good results in most cases but tends to reach local minima. However those results strongly depend on initial estimates of parameters provided, which is an important drawback, because in some cases finding the solution without well-chosen initial estimates can be difficult. The Authors tested those tuning rules on three different theoretical plants; first-order, second order and fractional-order. The experiment proved the usefulness of these rules and at the same time showed that, as in the case of the classical Z-N method, the resulting settings offer inferior performance to the one sought but can be applied even without knowledge of the plant model.

The last approach presented in [18] focuses on solely optimization algorithms. The authors considered BLDC (brushless direct-current) motor as a control plant and used PID and FOPID controller tuned with multiple different methods to test the performance of such system. They used genetic algorithm, fuzzy logic, Grey Wolf Algorithm, Artificial bee colony algorithm, Neural-networks and more, with all of them using the same conditions and cost functions. The most popular cost functions are defined with integral indexes, for example, Integral Absolute Error (IAE), Integral of Time Multiplied Squared Error (ITSE), Integral of Time Multiplied Absolute Error (ITAE). Each algorithm generated a certain controller set vector and then determined the value of the cost function, which it minimized at the next iteration. The main difference between the aforementioned solutions was the difference in the operation of the algorithm's core, which handled the generation and updating of new set vectors. According to the results obtained by the authors, all of the considered methods can be successfully applied to determine optimal setting for controller, which proved that optimization algorithms are viable option as tuning method.

4. IMPLEMENTATION FEASIBILITY STUDY

Theoretical analyses of algorithms using non-integer-order controllers for process control purposes are becoming an increasingly common phenomenon and indicate the superiority of such controllers over classical solutions [19, 20, 21, 22, 23]. However, the part related to hardware implementation is currently at an early stage, and publications related to it, as well as available tools, are few. Delving into the currently occurring attempts to implement the aforementioned algorithms, it can be seen that they are mostly focused on problems related to electric motors, in particular DC motors [10, 24], PMSM motors [25] and servo motors [11, 26]. There

have also been attempts to control such objects as a PEM Fuel Cell [6], a heater [7], an Industrial boiler burning system [8], a water tank [9], as well as a simulation model of plant-coupled fluid tanks tested using a microcontroller [27]. The main sofware used to create and operate the algorithms were MATLAB/Simulink and Labview. On the other hand, the hardware part was mainly implemented using DAQ boards, PCI board or PLC, with the addition of real-time control libraries.

4.1. Available toolboxes

Compered to Labview, MATLAB has a wider range of available tools. In MATLAB's add-ons library there are some easy-to-use toolboxes, which provide complete structures and functions for fractional calculus and control, based on different definitions or approximations. Among them, the most extensive can be distinguished: the FOMCON toolbox [28, 29, 30, 31], FOTF toolbox [32] and Ninteger toolbox [33]. The FOMCON toolbox is the most complex available tools as of today, it includes many useful easy-to-use blocks such as complete FOPID Controller, fractional integrator/derivative, fractional transfer function both in continues and discrete-time domain and many more. FOMCON can be used for design, analysis, simulation and control of fractional-order systems. The fractional-order elements are approximated using Oustaloup's recursive filters, described in Section 3.2. It also comes with controller tuning under performance and robustness specifications. However, its most significant advantage lies in providing a comprehensive, end-to-end framework that facilitates the seamless transition from fractional-order system model to a fully implementable control solution. FOTF toolbox is less complex than FOMCON but still provides all the basic blocks; fractional operator, Caputo operator, Riemann-Liouville operator, FOPID Controller and FOTF Model.

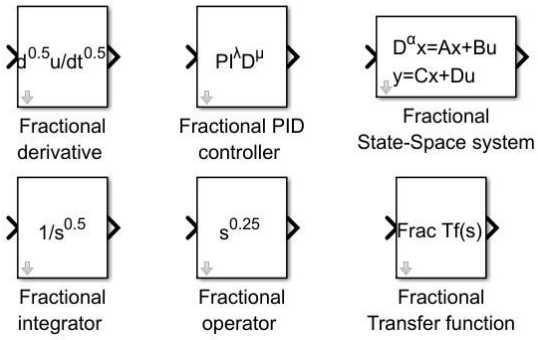

Fig. 4.1. Example of universal blocks from FOMCON toolbox

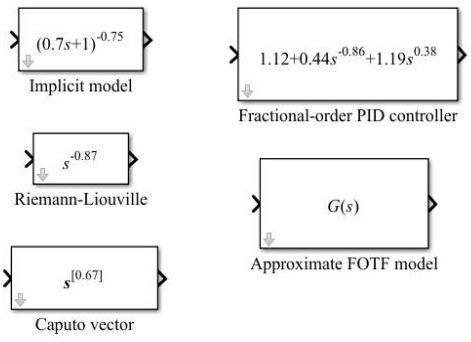

Fig. 4.2. Example of universal blocks from FOTF toolbox

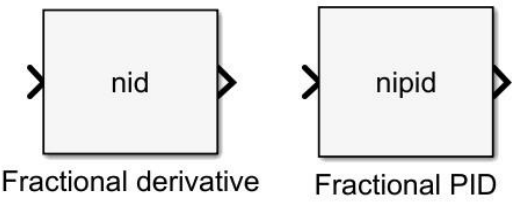

Fig. 4.3. Example of universal blocks from Ninteger toolbox

The Ninteger toolbox has a wide range of available functions with focus on CRONE Controllers and approximations but comes with only two universal block structures: Fractional derivative and FOPID controller. Figures 4.1, 4.2 and 4.3 show examples of universal blocks from the mentioned toolboxes. Those toolboxes are also suitable for hardware implementation.

4.2. Real-Time implementation

For real-time implementation of fractional-order algorithms, a number of tools can be highlighted. There are MATLAB libraries such as HDL Coder for DAQ boards with FPGA modules or FPGA and ASIC standalone systems and PLC Coder for PLC controllers. Both of those libraries are toolsets to automatically generate code from the simulink graphical interface for a particular target, for example, VHDL/Verilog for FPGAs and ST and Ladder Logic for PLCs, and allow verification of code operation, as well as providing support for external interfaces. However, these libraries support a limited number of hardware platforms, so one must verify compatibility with specific hardware before deciding to use them.

There is also third-party software interfacing with MATLAB available commercially, such as QUARC and dSPACE real-time software. These software solve the problem of handling embedded systems as a target and support the design and implementation of complex algorithms. Together with the software, it is possible to use dedicated measurement boards so that all control and measurements can be performed by a single unit fully compatible with Simulink. Within the scope of this work, the QUARC software kit was used together with a measurement board with an FPGA module, acting as an external target in the form of NI-myRIO-1900. Most of the measurement boards from National Instruments are compatible with the QUARC software; hence, the NI-PCI-6221 card could also be used. It is worth mentioning that the previously presented equation describing the FOPID controller (eq. (3.1)) in the case of practical implementation often requires consideration of saturation and anti-windup algorithms.

5. FRACTANCE DEVICES

A device demonstrating behavior governed by fractional-order dynamics is termed a fractance [34, 35]. The fractance devices can be classified by three basic types: domino ladder circuit network, a tree structure of electrical elements and net-grid network. Each is built with infinitely many resistors R and capacitors C connected in series or parallel. As part of the study, it was decided to consider the case of a domino ladder system described as a Cauer type I structure, shown in Figure 5.1. Since there is no phenomenon of infinity in reality, a fractance with dynamics corresponding to a non-integer order α system, can be approximated by finite number of

DOI 10.2478/ama-2025-0031

elements using truncated continued fraction expansion (CFE). Using the aforementioned CFE method, the impedance of domino ladder system can be described as: [34]

$$Z_{DL}(s) = R_0 + \cfrac{1}{C_1 s + \cfrac{1}{R_1 + \cfrac{1}{C_2 s + \cdots \cfrac{1}{R_{n-1} + \cfrac{1}{C_n s + \cfrac{1}{R_n}}}}}} \approx \frac{1}{C_\alpha s^\alpha} \tag{5.1}$$

where, C_α - is pseudo-capacity of the system, R_n is resistance value and C_n is capacitance value. The circuit RC_α in Figure 5.1, can be described using Kirchhoff's second law as:

$$u_{in}(t) = u_r(t) + u_{out}(t),\ t \geq 0, \tag{5.2}$$

where $u_R(t)$ is voltage of resistor R, $u_{in}(t)$ is source voltage e and $u_{out}(t) = V_0 - V_-$ is voltage on the fractional-order element.

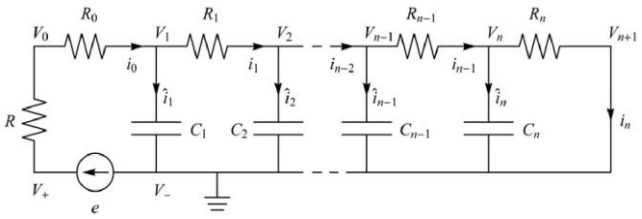

Fig. 5.1. Distribution of potentials and currents in the ladder system [34]

Considering that

$$i_0(t) = C_\alpha\, {}_a^C D_t^\alpha u(t), \tag{5.3}$$

equation eq. (5.2) can be expressed as

$$u_{in}(t) = RC_\alpha\, {}_a^C D_t^\alpha u_{out}(t) + u_{out}(t),\ t \geq 0, \tag{5.4}$$

which can then be converted to

$${}_a^C D_t^\alpha u_{out}(t) = -\frac{u_{out}(t)}{RC_\alpha} + \frac{u_{in}(t)}{RC_\alpha},\ t \geq 0, \tag{5.5}$$

after applying the Laplace transform, the following was obtained

$$Z(s) = \frac{U_{out}(s)}{U_{in}(s)} = \frac{1}{RC_\alpha s^\alpha + 1} \tag{5.6}$$

6. SIMULATION ANALYSIS

6.1. Toolbox comparison

The domino ladder system from Figure 5.1 was recreated in Simulink using values specified in [34] for α = 0.60 with Simscape electrical library as shown in Figure 6.1. That is because such system can effectively be used as a base sample for comparison with models created with considered toolboxes.

Using eq. (5.6) and FOMCON, FOTF and NINTEGER Toolboxes several models were created with approximation order of 5 and frequency range [0.001;1000]. All of the considered models were then supplied with a voltage step signal of 10 V and their response was compared with domino ladder system response. This step is essential to verify that all fractional integrators provided by the toolboxes perform as expected and to identify the most suitable implementation before applying them in control scenarios. The responses of those models are shown in Figure 6.2, where the difference between them is apparent, but appears to be insignificant.

Hence, in addition, a correlation showing the relative error between the responses of the models and the domino ladder system over time was determined, which can be seen in Figure 6.3.

Analyzing the results shown in Figure 6.3, it turns out that the best approximation of the non-integer-order element was obtained for the FOMCON toolbox. Therefore, FOMCON Toolbox will be used for simulation and practical implementation.

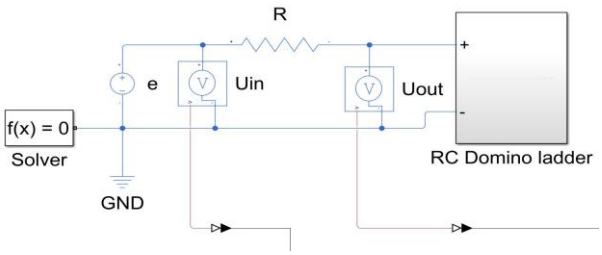

Fig. 6.1. Domino ladder system created in Simulink for α= 0.60

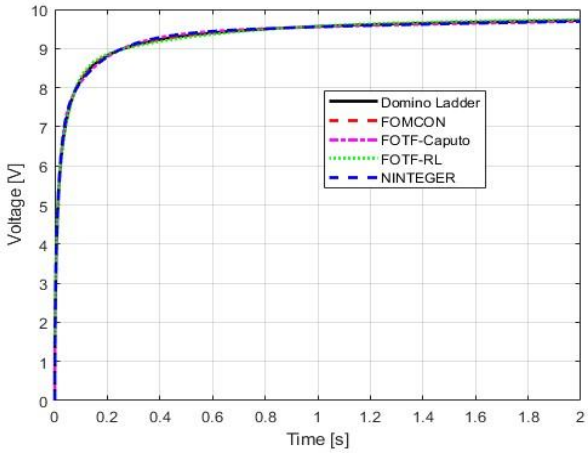

Fig. 6.2. Comparison of the responses of the considered fractional-order models

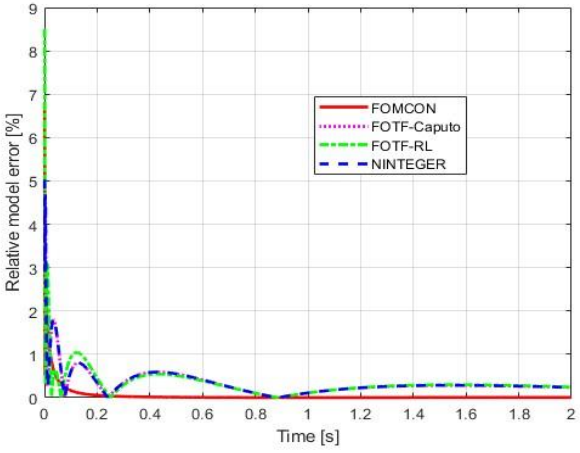

Fig. 6.3. Relative error of the considered fractional-order models

6.2. Second order system

Now an example implementation of a fractional-order algorithm using the aforementioned tools will be presented, where a very simple system without delays will be taken as the control plant in order

to show the required operations. Note, however, that in reality many systems have delays, but the approach will remain the same. Now let's consider second-order control plant which can be described as

$$G(s) = \frac{U_{out}(s)}{U_{in}(s)} = \frac{K}{(sT_1+1)(sT_2+1)}. \quad (6.1)$$

When K = 1, T1 = 1.5 and T2 = 2.5 the transfer function eq. (6.1) of plant takes form

$$G(s) = \frac{1}{3.75s^2+4s+1}. \quad (6.2)$$

Based on equation (6.2), a simulation was carried out using both PID and FOPID controllers, where each controller was implemented according to equation (3.1). For the PID controller, the parameters were set to $\lambda = 1$ and $\mu = 1$, while for the FOPID controller, the values were selected within the fractional ranges $\lambda \in (0,1)$ and $\mu \in (0,1)$.

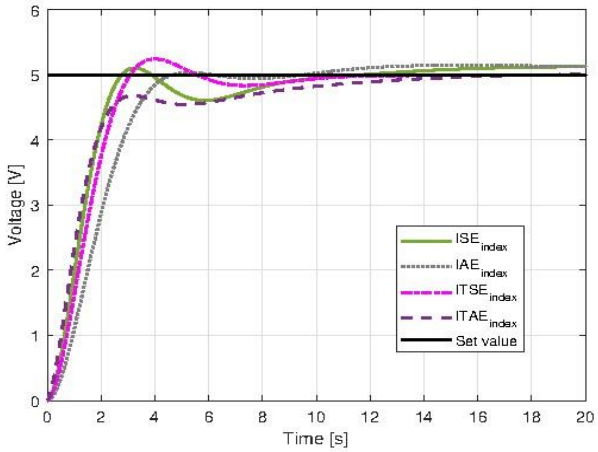

Fig. 6.4. Comparison of the response of the system with a FOPID controller tuned by considering different integral indices

As for the tuning method, Modified Grey Wolf Optimization algorithm [18, 36] was implemented and both controllers were tuned with the same conditions and constraints. The key distinction lies in the number of optimized parameters: the PID controller required tuning of three parameters (kp, ki, kd), whereas the FOPID controller involved five parameters (kp, ki, kd, λ, μ).The cost function eq. (6.3) consisted of the cost of deviation (CF_E) and the cost of control (CF_U) with weights of 400:20. Considering the control effort in the optimization process helps to obtain controller settings that can be realistically implemented in hardware, as the resulting control signal stays within the limits supported by the measurement card, e.g. ±10V. The cost of deviation can be determined using different integral indices: ISE, IAE, ITSE or ITAE, while the cost of control was determined as the square of the error. Depending on the selected performance index, the optimization algorithm emphasizes different characteristics of the system response. Therefore, a separate controller tuning procedure was performed for each of the specified indices to examine how the choice of a particular criterion influences the final outcome of the algorithm. The algorithm yielded four sets of settings for the PID and FOPID controller, using the aforementioned indices, 40 iterations and 20 searching agents. It is important that the optimization process is carried out for a higher setpoint than the one envisaged in the target implementation. The closed-loop response of the system (6.2) with FOPID control is shown in Figure 6.4. The ITSE criterion provided the best performance, and thus its associated controller settings were adopted for implementation. The cycle execution time was 0.002 s for both controllers.

Cost function = 400·*CF_E* +20·*CF_U* (6.3)

7. HARDWARE IMPLEMENTATION

7.1. Implementation procedure

The hardware unit supporting the control algorithm is a measurement board NI-myRIO-1900. Its operation and control have been implemented using QUARC software, which provides the blocks shown in Figure 7.1. The first of these (HIL Initialize) is the hardware configuration block, where all the parameters of the system are specified, including the number of inputs/outputs and their channel numbers, measurement ranges, encoder inputs, etc. Further on, you can see the analog input and output blocks of the card, as well as the block that allows you to save data to a file (To Host File). Analog laboratory model was used to simulate the operation of the system eq. (6.2).

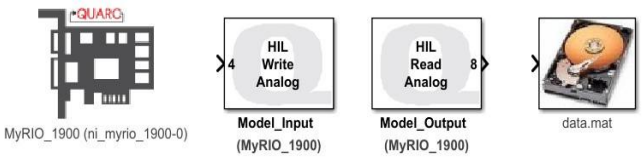

Fig. 7.1. Example of QUARC blocks for real-time implementation

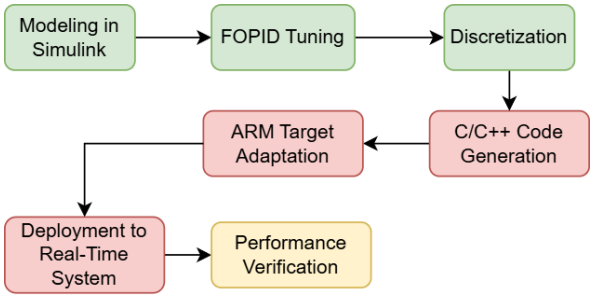

Fig. 7.2. A flow chart of the procedure for implementing the control algorithm

The process of implementing the algorithm from Section 6.2. into a real system is shown in Figure 7.2. The first three steps are associated with Matlab Simulink software, while the next four are implemented by the QUARC add-on.

1. Modeling in Simulink – design the chosen control algorithm, in this case a PID/FOPID controller in Simulink. It is important to use blocks from the QUARC library, from Figure 4.1 and Figure 7.1. Remember to define the I/O along with the board configuration via the HIL Initialize block.
2. FOPID Tuning – input the designated controller settings – kp, ki, kd, λ, μ.
3. Discretization – as the algorithm is ultimately intended for a real system, it should be discretized with sample time Ts. In this case, Ts was taken as 0.002s.
4. C/C++ Code Generation – compile the discretized model to C/C++ using QUARC's auto-code tools.
5. ARM Target Adaptation – modify generated code for ARM architecture compatibility (e.g., myRIO-1900).

DOI 10.2478/ama-2025-0031

6. Deployment to Real-Time System – upload the executable to the myRIO-1900 real-time target.
7. Performance Verification – validate real-time operation against design requirements (e.g., step response, stability).

It is critical to select the correct target type in the QUARC compiler options during the initial model preparation stage (Step 1), as Steps 4–6 are automatically executed by QUARC during compilation based on the specified target. For this implementation, the quarc_linux_rt_armv7 target was chosen, as it provides a generic real-time framework for ARM-based systems, including the myRIO-1900 board.

Due to the interferences introduced by the control plant, a low-pass filter described as eq. (7.1), was applied to its output to negate its effect on system operation.

$$G(s) = \frac{75}{s+75} \tag{7.1}$$

7.2. Experimental Results

The system from section 6.2 has been implemented with a procedure from Figure 7.2. into myRIO-1900 using the PID and FOPID controller parameters optimized for the ITSE criterion. Figure 7.3 presents the closed-loop responses of both the physical plant and simulated system, while Figure 7.4 displays their corresponding control signals.

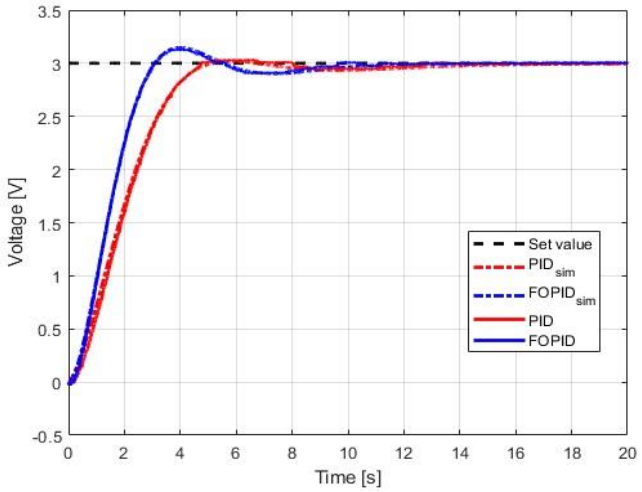

Fig. 7.3. Comparison of the simulation and real control plant responses for PID and FOPID controller

Analyzing the results shown in Figures 7.3 and 7.4, it is evident that the experimental results coincided with the simulation results. The values of the cost function obtained in both the simulation and the experiment, along with the selected values of the controller settings, are summarized in Table 1. Clearly the FOPID controller scored about 30% better in terms of control deviation and about 6% better in terms of control cost than the PID controller in both the simulations and the experiment. The FOPID controller demonstrated superior performance with a 16% lower RMSE (Root Mean Square Error) value compared to the conventional PID controller, further validating its effectiveness for the given control application. The experimental results also yielded better results in terms of control deviation than the simulation with comparable value of control signal. In addition, in the waveform of the control signal in Figure 7.4, it can be seen that the FOPID Controller handled the control signal better. It is also worth mentioning that control signal from FOPID controller was more resistant to interference than PID controller, which is visible in Figure 7.4.

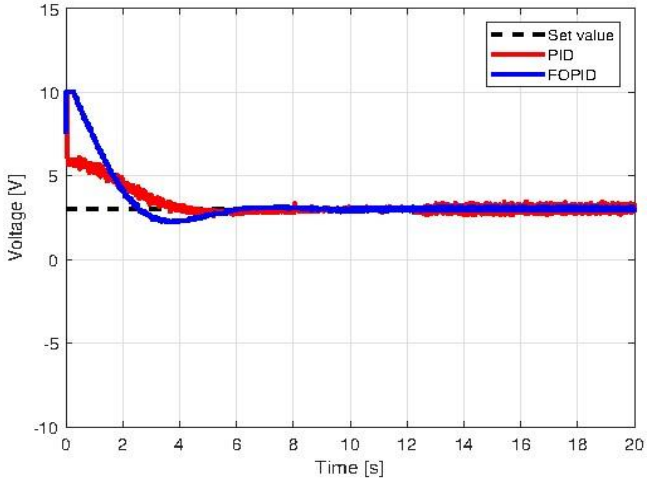

Fig. 7.4. Comparison of control signals from PID and FOPID controller from the experiment

Tab. 1. Controller parameters and cost function for the system

	PID Controller	*FOPID Controller*
K_p	1,8902	2,5043
K_i	0,4785	0,6644
λ	1,0000	1,0060
K_d	0,4370	0,7156
μ	1,0000	0,5027
CF_E_{SIM}	12,4389	8,0557
CF_E_{REAL}	11,0435	7,5231
CF_U_{SIM}	318,4723	300,3850
CF_U_{REAL}	328,7697	306,1467
RMSE	0,7948	0,6858

8. CONCLUDING REMARKS

In this paper, a method for design and implementation of fractional-order controller for both simulation and experiment were presented. The analysis of currently existing solutions made it possible to identify current trends in the tuning of FO controllers and showed the superiority and potential of FOPID controllers over their classical counterparts. The available tools for working with fractional-order calculus were presented, and the FOMCON Toolbox was clearly identified as the current best solution. Then the physical implementation of the fractional-order element is shown, along with an example implementation procedure for a given model of a control object. It turns out that the process of tuning FOPID controllers is relatively lengthy and requires appropriate definition of search ranges and cost functions to enable a smooth transition from the simulation environment to the real one. The solution shown can

also be easily applied to other control objects and hardware platforms. Further research will focus on switched systems, which will eventually be enriched with fractional-order controllers.

REFERENCES

1. Kaczorek T, Rogowski K. Fractional Linear Systems and Electrical Circuits. Białystok Poland. Springer; 2015.
2. Tepljakov A. Fractional-order Modeling and Control of Dynamic Systems. Tallinn Estonia. Springer; 2017.
3. Luo A, Sun J. Complex Systems – Fractionality. Time delay and Synchronization. Springer; 2012.
4. Petras I. Fractional-Order Nonlinear Systems - modeling, analysis and simulation. Springer; 2011.
5. Das S, Pan I. Fractional Order Signal Processing: Introductory Concepts and Applications. Springer; 2012.
6. Lu X, Miao X, Xue Y, Deng L, Wang M, GU D, et al. Dynamic modeling and fractional order PIλDμ control of PEM fuel cell. Int J Electrochem Sci. 2017;12(8):7518–36. doi:10.20964/2017.08.12
7. Zamojski M. Implementation of fractional order pid controller based on recursive Oustaloup'e filter. In: 2018 International Interdisciplinary PhD Workshop (IIPhDW). Świnoujście Poland. 2018; 414–7. doi:10.1109/I-IPHDW.2018.8388402
8. He Y, Gong R. Application of fractional-order model reference adaptive control on industry boiler burning system. In: 2010 Int Conf Intell Comput Technol Autom. Changsha China. 2010; 750–3. doi:10.1109/ICICTA.2010.59
9. Bhambhani V, Chen Y. Experimental study of fractional order proportional integral (FOPI) controller for water level control. In: 2008 47th IEEE Conf Decision Control. Cancun, Mexico. 2008; 1791–6. doi:10.1109/CDC.2008.4739341
10. Tepljakov A, Gonzalez E, Petlenkov E, Belikov J, Monje C, Petráš I. Incorporation of fractional-order dynamics into an existing PI/PID DC motor control loop. ISA Trans. 2016;60:262–73. doi:10.1016/j.isatra.2015.11.012
11. Tepljakov A, Petlenkov E, Belikov J, Astapov S. Tuning and digital implementation of a fractional-order PD controller for a position servo. Int J Microelectron Comput Sci. 2013;4(3):116–23.
12. Sun J, Wang C, Xin R. Design of fractional order proportional differentiation controller for second order position servo system. In: 2018 Chinese Control And Decision Conf (CCDC). Shenyang China. 2018; 5939–44. doi:10.1109/CCDC.2018.8408171
13. Kadiyala V, Jatoth R, Pothalaiah S. Design and implementation of fractional order PID controller for aerofin control system. In: 2009 World Congr Nature Biol Inspired Comput (NaBIC). Coimbatore India. 2009; 696–701. doi:10.1109/NABIC.2009.5393470
14. Xu X, Lu Y, Vogel-Heuser B, Wang L. Industry 4.0 and industry 5.0—inception, conception and perception. J Manuf Syst. 2021;61:530–5. doi:10.1016/j.jmsy.2021.10.006
15. Shah P, Agashe S. Review of fractional PID controller. Mechatronics. 2016;38:29–41. doi:10.1016/j.mechatronics.2016.06.005
16. Kaczorek T. Fractional descriptor observers for fractional descriptor continuous-time linear system. Arch Control Sci. 2014;24:27–37. doi:10.2478/acsc-2014-0002
17. Valerio D, Costa J. Ziegler-Nichols type tuning rules for fractional PID controllers. In: ASME 2005 Int Des Eng Tech Conf Comput Inf Eng Conf; 2005. doi:10.1115/DETC2005-84344
18. Kumarasamy V, Ramasamy VKT, Chandrasekaran G, et al. A review of integer order PID and fractional order PID controllers using optimization techniques for speed control of brushless DC motor drive. Int J Syst Assur Eng Manag. 2023;14:1139–50. doi:10.1007/s13198-023-01952-x
19. Rastogi P, Chatterji S, Karanjkar D. Performance analysis of fractional-order controller for pH neutralization process. In: 2012 2nd Int Conf Power Control Embedded Syst (ICPCES). Allahabad, India. 2012; 1–6. doi:10.1109/ICPCES.2012.6508116
20. Petras I. Modeling and numerical analysis of fractional-order Bloch equations. Comput Math Appl. 2011;61(2):341–56. doi:10.1016/j.camwa.2010.11.009.
21. Bai L, Xue D. Universal block diagram-based modeling and simulation schemes for fractional-order control systems. ISA Trans. 2018;82:153–62. doi:10.1016/j.isatra.2017.04.018
22. Sun X, Yan G, Zhang B. The simulation analysis for a kind of fractional order Kalman estimator. Procedia Comput Sci. 2017;111:308–14. doi:10.1016/j.procs.2017.06.028
23. Dzielinski A, Sierociuk D. Simulation and experimental tools for fractional order control education. IFAC Proc. 2008;41(2):11654–9. doi:10.3182/20080706-5-KR-1001.01975
24. Kanzari B, Taieb A, Chaari A. Fractional modeling of the speed of a DC motor and control with FOPID controller. In: 2023 9th Int Conf Control, Decision Inf Technol (CoDIT). Rome Italy. 2023; 2085–90. doi:10.1109/CoDIT58514.2023.10284258
25. Thakar U, Joshi V, Vyawahare V. Design of fractional-order PI controllers and comparative analysis of these controllers with linearized, nonlinear integer-order and nonlinear fractional-order representations of PMSM. Int J Dyn Control. 2017;5:187–97. doi:10.1007/s40435-016-0243-0.
26. Barbosa R, Machado J, Jesus I. On the fractional PID control of a laboratory servo system. IFAC Proc. 2008;41(2):15273–8. doi:10.3182/20080706-5-KR-1001.02583
27. Tepljakov A, Petlenkov E, Belikov J. Embedded system implementation of digital fractional filter approximations for control applications. In: 2014 Proc 21st Int Conf Mixed Des Integr Circuits Syst (MIXDES). Lublin Poland. 2014; 441–5. doi:10.1109/MIXDES.2014.6872237
28. Tepljakov A, Petlenkov E, Belikov J. FOMCON: A MATLAB toolbox for fractional-order system identification and control. Int J Microelectron Comput Sci. 2011;2:51–62.
29. Tepljakov A, Petlenkov E, Belikov J. Closed-loop identification of fractional-order models using FOMCON toolbox for MATLAB. In: 2014 14th Biennial Baltic Electron Conf (BEC). Tallinn Estonia. 2014; 213–6. doi:10.1109/BEC.2014.7320594
30. Tepljakov A, Petlenkov E, Belikov J. A flexible MATLAB tool for optimal fractional-order PID controller design subject to specifications. In: Proc 31st Chinese Control Conf. Hefei China. 2012; 4698–703.
31. Tepljakov A, Petlenkov E, Belikov J. FOMCON: Fractional-order modeling and control toolbox for MATLAB. In: Proc 18th Int Conf Mixed Des Integr Circuits Syst (MIXDES). Gliwice Poland. 2011; 684–9.
32. Li T, Xue D, Cui X. A tutorial on a universal blockset for fractional-order systems. In: 2021 33rd Chinese Control and Decision Conf (CCDC). Kunming China. 2021; 1210–6. doi:10.1109/CCDC52312.2021.9602256
33. Valerio D, Costa J. Ninteger: A non-integer control toolbox for MATLAB. In: Proc 1st IFAC Workshop Fractional Differentiation Appl; 2004.
34. Piotrowska E. Analiza obwodów elektrycznych zawierających elementy opisane pochodną o różnych rzędach niecałkowitych [PhD dissertation]. Białystok Poland: Politechnika Białostocka; 2021.
35. Petras I, Sierociuk D, Podlubny I. Identification of parameters of a half-order system. IEEE Trans Signal Process. 2012;60(10):5561–6. doi:10.1109/TSP.2012.2205920
36. Yazici I, Yaylaci E. Modified grey wolf optimizer based MPPT design and experimentally performance evaluations for wind energy systems. Eng Sci Technol Int J. 2023;46. doi:10.1016/j.jestch.2023.101520

This research was carried out in the framework of the grant no. 2022/45/B/ST7/03076 financed by the National Science Center in Poland.

Paweł Kieczmerski: https://orcid.org/0009-0008-5029-3792