Event-based fractional order control

Isabela Birs a,b,c, Ioan Nascu a, Clara Ionescu b,c, Cristina Muresan a,*

a Technical University of Cluj-Napoca, Automation Department, Memorandumului 28, Cluj-Napoca, Romania
b Ghent University, Department of Electromechanical, Systems and Metal Engineering, Research Lab on Dynamical Systems and Control, Tech Lane Science Park 125, 9052 Ghent, Belgium
c Flanders Make, EEDT - Decision and Control Group, Tech Lane Science Park 131, 9052 Ghent, Belgium

GRAPHICAL ABSTRACT

The present study provides a generalization of event-based control to the field of fractional calculus, combining the benefits brought by the two approaches into an industrial-suitable control strategy. During recent years, control applications based on fractional order differintegral operators have gained more popularity due to their proven superior performance when compared to classical, integer order, control strategies. However, the current industrial setting is not yet prepared to fully adapt to complex fractional order control implementations that require hefty computational resources; needing highly-efficient methods with minimum control effort. The solution to this particular problem lies in combining benefits of event-based control such as resource optimization and bandwidth allocation with the superior performance of fractional order control. Theoretical and implementation aspects are developed in order to provide a generalization of event-based control into the fractional calculus field. Different numerical examples validate the proposed methodology, providing a useful tool, especially for industrial applications where the event-based control is most needed. Several event-based fractional order implementation possibilities are explored, the final result being an event-based fractional order control methodology.

ARTICLE INFO

Article history:
Received 16 April 2020
Revised 3 June 2020
Accepted 27 June 2020
Available online 3 July 2020

2010 MSC:
26A33
60G22
90C32

Keywords:
Fractional calculus
Fractional order control
Event-based control

ABSTRACT

The present study provides a generalization of event-based control to the field of fractional calculus, combining the benefits brought by the two approaches into an industrial-suitable control strategy. During recent years, control applications based on fractional order differintegral operators have gained more popularity due to their proven superior performance when compared to classical, integer order, control strategies. However, the current industrial setting is not yet prepared to fully adapt to complex fractional order control implementations that require hefty computational resources; needing highly-efficient methods with minimum control effort. The solution to this particular problem lies in combining benefits of event-based control such as resource optimization and bandwidth allocation with the superior performance of fractional order control. Theoretical and implementation aspects are developed in order to provide a generalization of event-based control into the fractional calculus field. Different numerical examples validate the proposed methodology, providing a useful tool, especially for industrial applications where the event-based control is most needed. Several event-based fractional order implementation possibilities are explored, the final result being an event-based fractional order control methodology.

© 2020 The Authors. Published by Elsevier B.V. on behalf of Cairo University. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

https://doi.org/10.1016/j.jare.2020.06.024

2090-1232/© 2020 The Authors. Published by Elsevier B.V. on behalf of Cairo University.
This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).
Introduction

Fractional calculus is a powerful tool with multiple applications in science and engineering. The main feature lies in the ability to accurately capture physical phenomena in a more natural and generalized manner than other available mathematical tools. Fractional applications spread through a multitude of fields such as physics, chemistry, mathematics, statistics, acoustics, engineering from both control and modeling point of views, providing excellent results through elegant mathematical solutions [1–4]. Its full potential is yet to be grasped by the scientific community through the multitude of papers emerging every year proving wide applicability of fractional calculus with uplifting results.

From the engineering perspective, fractional calculus has generalized the degrees of freedom of control possibilities and has revolutionized the modeling of memory based structures [5]. The ubiquity of the fractional order memory system gives a structure that is more concise in form, but encapsulates a great accuracy of complex dynamics [6]. The main contribution of fractionalization in the control field consists of the generalization of the popular Proportional Integral Derivative (PID) controller. The resulting strategy has been proven to be superior to its integer order instance. The obtained closed loop system featuring fractional order control has been proven to have increased time domain performance, stability, robustness and flexibility from the frequency response perspective [7–9]. Its diverse benefits contribute to the enhancement of multiple control possibilities such as the fractional order Internal Model Control (IMC) strategy, fractional order filter integration to available integer-order structures, fractional order Linear Quadratic Regulator (LQR), etc [10–13].

The present work focuses on introducing fractional calculus in a control branch that has not been previously explored from this perspective: the event-based control strategy. Event-based control is a relatively novel concept that triggers the computation of a new control signal through events, in comparison to classical controller implementations that trigger time based changes in the control signal [14,15]. The strategy is mostly beneficial for processes that have limited resources such as CPU, bandwidth allocation and reduced energy demand [16,17]. All of these features show the necessity of the event-based strategy in today’s industrial settings [18]. Furthermore, cloud control systems are becoming more popular, the industry being the main beneficiary of such an approach that externalizes automatic control, providing a versatile solution that could be easily debugged and adapted [19,20]. Event-based control combined with cloud implementations is another powerful tool for the future of industrial control. The aim of the study is to introduce all the benefits provided by fractional order control into the event-based strategy, offering multiple possibilities to exploit these advantages to benefit the current industrial need.

The contribution of the current work to the fractional calculus field is the engineering applicability of this numerical tool into event-based control strategies. The theoretical background for event-based fractional order controllers is proposed and several implementation options are explored. The method is validated for the general fractional order PID controllers using varied numerical examples. Furthermore, the proposed approach is compared with classical, non-event based, discrete-time approaches showing the efficiency of the methodology as well as the benefits regarding the control effort.

The paper is structured as follows: Section 2 presents a brief familiarization with fractional calculus in control applications as well as current trends in fractional order control implementations; Section 3 is focused on existing event-based control approaches and provides a solid motivation of its usage in industry; Section 4 details the proposed event-based fractional order control strategy, theoretical aspects and implementation guidelines; Section 5 presents four numerical examples centered around the fractional order PID controller implemented using the proposed event-based strategy; while Section 6 concludes the paper. In addition, Appendix A provides the event-based fractional order control algorithm in a programming fashion.

Fractional calculus in control applications

Fundamentals of fractional calculus

Fractional calculus is a generalization of differentiation to any arbitrary order. The limit that integration and differentiation orders are limited to integer numbers is removed, creating a new branch of mathematics known as non-integer (fractional) calculus. The differintegral operation is generalized into the fundamental operator $D_a^x$, where $a \in R$ is the fractional order [21].

The most common mathematical definition of the fractional order operator is given by Riemann–Liouville [22–24] as

$$sD_a^xM(t) = \int_a^t (t-\tau)^{x-1} M(\tau) d\tau,$$

(1)

where $D$ represents an integral for negative order and a derivative for the positive case from an arbitrary fixed base point $a$, and $\Gamma(z)$ is Euler’s Gamma function [25].

Another possibility to express the fractional differintegral operation is the Grunwald–Letnikov formula [25,26]

$$sD_a^xM(t) = \lim_{h \to 0} h^{-x} \sum_{0 \leq j < \infty} (-1)^j \binom{x}{j} M(t-jh),$$

(2)

where $\binom{x}{j}$ represents the fractional binomial coefficient.

Applying the Laplace transform on the fractional differintegral of order $a$ gives

$$L\{D_a^x f(t)\} = s^{-x} F(s)$$

(3)

and

$$L\{D_a^x f(t)\} = s^x F(s)$$

(4)

where $F(s) = L\{f(t)\}$ and $s$ the Laplace complex variable.

The connotations of (3) and (4) in automatic control are the extensions of the integer-order PID controller to its fractional order generalization denoted by

$$C(s) = k_p \left(1 + \frac{k_i}{s^\lambda} + k_ds^{\mu}\right)$$

(5)

where $k_p$ is the proportional gain, $k_i$ and $k_d$ are the integral and derivative terms, $\lambda$ is the fractional order of integration and $\mu$ is the fractional order of differentiation. For the particular case $\lambda = 1$ and $\mu = 1$, the controller from (5) denotes the classical, integer order, PID controller. As can be seen from the form of the fractional order PID controller, the fractional orders of $\lambda$ and $\mu$ lead to a more flexible structure than its integer order particularity. The PID generalization is capable of solving a wider and more restrictive set of performance specifications, having the shape of the popular PID controller, but with more power and pliability. Hence, a performing control option becomes available that proves better time response, increased stability, robustness and versatility [7,8,27,28].

Implementation possibilities of fractional order control

Analog implementation possibilities of this type of control are scarcely available, most of them based on fractance circuits [29]. Continuous controller implementation requires electronic components that are mostly suitable for integer order transfer functions. Some works such as [30–35] tackle analog fractional devices using
I. Birs et al. / Journal of Advanced Research 25 (2020) 191–203

The fifth order discrete time approximations from (7) and (10) based on the integer order Tustin formula should be sufficient for most applications. These approximations will be used further to directly map the fractional order $s^\alpha$ into the discrete-time domain with the purpose of creating the event-based fractional order control theory.

Event-based control

Context awareness

The fourth industrial revolution, known as Industry 4.0, integrates the latest scientific advances of the 21st century into the architecture of traditional manufacturing plants. State of the art smart manufacturing systems and technologies take the industrial setting a step further into the future. Human intervention is diminished by improved automation technologies featuring state machines, self-monitoring and abilities to perform machine to machine communication. The emerging Internet of Things (IOT) protocol is the principal constituent in the smart manufacturing trend, creating connected networks of distributed systems [47].

The new industrial cloud architectures are equipped with a wide range of sensors and actuators creating a widespread network that gathers and shares data. Decisions are autonomously taken by interpreting the available information [48]. However, an overflow of unnecessary data leads to an overburden of the bandwidth and wireless network sharing. Introducing the context awareness modules into the smart manufacturing process enables the interpretation and filtering of the gathered information as well as performing decisions regarding its relevance [49]. The main idea is to process data in the lower levels of the system and forwarding the contextual information to the upper layers of the architecture. There are four main layers of the context management, as presented in Fig. 1.

Each objective of the system is analyzed and a context map is creating based on regression data. The context map determines various sets of input and output pairs that are particularly useful in reaching the objective (with green), pairs that are layabout for reaching the goal (with orange) or the pairs that hinder the main objective (with red). The next step involves an analysis of the process with respect to the previously defined sets and identifying events. These events will ultimately lead to a new context, with different pairs than the previous, by triggered the computation of a new set of control parameters in order to reach the main objective.

From the control engineering point of view, the natural solution of the context awareness paradigm lies in the event-based control methodology. The system’s context is mapped based on some predefined conditions for reaching a certain objective, specific to the controlled process.

Even if traditional sampled data control, known as Riemann sampling, is the standard tool for implementing computer control, there are severe difficulties when dealing with systems having multiple sampling rates or systems with distributed computing [50]. With multi-rate sampling, the complexity of the system depends critically on the ratio of the sampling rates. For distributed systems, the theory requires that the clocks are synchronized. However, in networked distributed systems, current research revolves around solving problems such as sampling jitter, lost samples and delays on computer controlled systems [51]. Event based sampling, also known as Lebesgue sampling, is a context aware
alternative to periodic sampling [52]. Signals are then sampled only when significant events occur, such as when a measured signal exceeds a limit or when an encoder signal changes. Event based control has many conceptual advantages. Control is not executed unless it is required, control by exception. For a camera based sensor it could be natural to read off the signal when sufficient exposure is obtained. Event based control is also useful in situations when control actions are expensive, for example when controlling manufacturing complexes, and when it is expensive to acquire information like in computer networks [53].

A severe drawback with event based systems is that very little theory is available. All sampled systems, periodic as well as event based, share a common property that the feedback is intermittent and that control is open loop between the samples. After an event, the control signal is generated in an open loop manner and applied to the process. In traditional sampled-data theory the control signal is simply kept constant between the sampling instants, a scheme that is commonly called a zero order hold (ZOH). In event based systems, the generation of the open loop signal is an important issue and the properties of the closed loop system depends critically on how the signal is generated [53]. There has not been much development of theory for systems with event based control, compared to the widely available strategies developed for periodic control.

Mathematical analysis related to the efficiency of the Lebesgue sampling algorithms has been realized by [53]. The study compares the performance of Riemann sampling to event based sampling and shows that the latter gives a better performance, even if the variable sampling theory is still in its infancy.

Industrial benefits

Successful industrial event based applications have been reported during recent years in works such as [14,54,55] spanning on a manifold of industrial processes from the simple tank level control [56] to microalgae cultures in industrial reactors [57,58].

Event-based or event-driven control is a strategy based on variable sampling of the control action. The concept is best explained in contrast to the classical, non-event based, discrete-time implementation. The latter implies the realization of the discrete-time controller using a control signal computed every $T_s$ seconds, without exceptions. For step references, the control signal varies while the system is in the transient regime, afterwards the control variable is constant for most processes. The only exception to this is the presence of unwanted disturbances, when the control action tries to bring the system back to the steady-state regime. Hence, the control variable changes during the transient regime until the process variable reaches the desired reference or when rejecting disturbances, while for the other time slots it is constant. However, discrete-time control implementations compute a new value for the control signal every $T_s$ seconds. It is obvious that the approach is inefficient and that the control effort can be reduced to the periods of time when the process variable is outside the steady-state boundaries. The efficiency problem is solved in an elegant manner by the introduction of event-based control approaches that allow the computation of a new control signal with variable sampling, removing the stress on the control circuit for unnecessary computations.

Event-based control introduces multiple benefits such as CPU resource optimization by limiting the number of times the control signal is computed. This allows the sharing of the CPU with other processes in distributed architectures and also improves energy efficiency in a control scheme [14,59].

Another trend in industrial control [19] is industrial cloud computing. The concept externalizes the process computer to a cloud-based implementation. For complex control necessities, the control signal can be computed into the cloud by a powerful computer, shared with other processes. This approach reduces the cost and removes the necessity of powerful process computers. Event-based strategies are useful in this scenario through the reduced bandwidth allocation [19,60].

Basic principles

Fig. 2. presents a schematic of the event-based implementation concept.

The process data is acquired with the sampling time $h_{\text{min}}$, which is a constant value, chosen in a similar fashion as the sampling period in classical discrete-time implementations. The data is transferred into the event detector, whose sole purpose is to determine whether to trigger an event. The event detector can be customized with any rule that optimizes the control process [18,61].

One of the most popular event triggering rule is focused on the error variation between a predefined interval $[-\Delta_e, \Delta_e]$.

$$| e(t) - e(t - h_{\text{act}}) | \geq \Delta_e, \quad (12)$$

where $h_{\text{act}}$ is the elapsed time since the triggering of the previous event, $e(t - h_{\text{act}})$ is the error at the previous event triggering moment and $e(t)$ is the error at the current moment. Note that the error signal at the present moment $t$ is computed using $e(t) = Y(t) - Y_{sp}(t)$, where $Y(t)$ is the current process variable and $Y_{sp}(t)$ is the setpoint value.

Another popular event detection condition is the maximum allowed time between two consecutive events, known throughout literature as the safety condition. The maximum time is denoted by $h_{\text{max}}$, while the elapsed time since the previous event is $h_{\text{act}}$ [62]. The safety condition can be expressed as

$$h_{\text{act}} \geq h_{\text{max}}. \quad (13)$$

The triggering of an event employs the control input generator to compute a new value for the control signal. The control input gener-
Event-based integer order control

A brief exemplification of the discrete-time model of the controller inside the control input generator is realized for the integer order PID controller. The transfer function is provided in (5) for the particularity \( \lambda = \mu = 1 \). The PID control signal in the Laplace domain can be computed using

\[
U(s) = k_p(Y_p(s) - Y(s)) + \frac{k_p k_i}{s}(Y_p(s) - Y(s)) + k_p k_d(s)(Y_p(s) - Y(s)),
\]

where \( \beta \) and \( \gamma \) represent weighting factors of the setpoint value for the proportional and derivative actions.

Denoting \( e(t) = Y_p(t) - Y(t) \), \( e_c(t) = \beta Y_p(t) - Y(t) \) and \( e_d(t) = \gamma Y_p(t) - Y(t) \) and splitting the control law into three individual signals: proportional \( u_p(s) \), integral \( u_i(s) \) and derivative \( u_d(s) \) gives

\[
\begin{align*}
    u_p(s) &= k_p e_p(s), \\
    u_i(s) &= \frac{k_i}{s} e(s), \\
    u_d(s) &= k_d k_c e_d(s).
\end{align*}
\]

The discrete-time PID control law can be computed as the sum of the three individual signals into their discrete-time approximation. Using the Tustin method from (6) where \( \alpha = 1 \) and replacing \( s = \frac{2}{T_s} \frac{1-z^{-1}}{1+z^{-1}} \) in the individual components from (15) gives

\[
\begin{align*}
    u_p(k) &= k_p e_p(k), \\
    u_i(k) &= h_{\text{inc}} \left( k_p e(k) + e(k-1) + u_i(k-1) \right), \\
    u_d(k) &= \frac{k_d k_c}{T_s} e_d(k) - e_d(k-1) - u_d(k-1).
\end{align*}
\]

where \( u_i(k-1) \) is the previously computed integral component and \( u_d(k-1) \) is the previous derivative component. A previous computation refers to the moment of the last event. Note that the discrete sampling time, \( T_s \), has been replaced by \( h_{\text{inc}} \) which is the variable sampling time.

Eq. (16) gives the control law of the PID controller with variable sampling time. Following this model, other controllers such as the integer order IMC can be discretized and implemented inside the control input generator, as long as the sampling time is kept as a variable. In addition, the discretization method is not limited to Tustin, any viable discrete-time mapper can be used.

Event-based fractional order control

Proposed implementation methodology

The theoretical contribution of the present study is the generalization of the available event-based control strategy into the fractional order field. The challenge lies in the isolation of the sampling time and introducing its variability in the fractional order discrete-time approximation. There is not any specialized literature available that explores the possibilities of implementing fractional order controllers in the control input generator.

The development of the discrete fractional order control actions is based on fractional order direct discrete-time mappers. For an in depth analysis, both the Muir recursive and CFE discrete-time approximations from (7) and (10) will be used to compute discrete-time equivalent controllers. The aim in studying both discrete time mappers is to prove that any direct mapper whose sampling time can be isolated from the rest of the function can be successfully used in fractional order event-based implementations.

The event-based implementation of the fractional order PID controller from (5) with \( \lambda, \mu \in (0, 1) \) will be exemplified further. Following this model, any fractional order controller can be discretized and implemented in the fractional order control signal generator.

The fractional order control law is written as a sum of the three individual components as \( U_f(s) = u_p(s) + u_i(s) + u_d(s) \), with \( u_p(s), u_i(s) \) and \( u_d(s) \) being the proportional, integer and derivative fractional order control signals. Hence, generalizing (15) gives

\[
\begin{align*}
    u_p(s) &= k_p e_p(s), \\
    u_i(s) &= \frac{k_i}{s} e(s), \\
    u_d(s) &= k_d k_c s^\alpha e_d(s).
\end{align*}
\]

Both direct discrete-time approximations based on the Muir recursion or on the Continued Fraction Expansion approximate the fractional order term \( s^\alpha \) to a transfer function of the form

\[
s^\alpha \approx \left( \frac{2}{T_s} \right)^\alpha \left[ \frac{a_n z^{-n} + a_{n-1} z^{-n-1} + \ldots + a_0}{b_0 z^{\alpha} + b_{-1} z^{\alpha-1} + \ldots + b_{-n}} \right] = \left( \frac{2}{T_s} \right)^\alpha N_\alpha(z^{-1}) / D_\alpha(z^{-1}).
\]

For the Muir recursion mapper of \( s^\alpha \) the polynomials are \( N_\alpha(z^{-1}) = A_\alpha(z^{-1}, x) \) and \( D_\alpha(z^{-1}) = A_\alpha(z^{-1}, -x) \), while for the CFE mapper the polynomials are \( N_\alpha(z^{-1}) = P_\alpha(x) \) and \( D_\alpha(z^{-1}) = Q_\alpha(x) \). The sampling time has been isolated from the rest of the model and will be treated as a variable in the event-based implementation. The \( N_\alpha(z^{-1}) \) and \( D_\alpha(z^{-1}) \) can be computed only once and will remain the same for every iteration through the control signal generator.

Furthermore, the fractional order terms will be treated separately and approximated to the discrete-time domain using the generalized formula from (18). The fractional order proportional
component is identical to the integer-order PID proportional term since it does not introduce any fractional order operation
\[ u_p(k) = k_p e_p(k). \]

The fractional order integral term can be rewritten \( \frac{1}{s} = \frac{1}{z^{1-\lambda}} \) such that the direct discretization of \( s^\lambda \) respects the \( c > 0 \) requirement. Hence, the discrete-time transfer function of the fractional order integral component is computed using the Tustin formula for the \( \frac{1}{s} \) term and a direct \( n^{th} \) discrete-time formula for the \( s^{1-\lambda} \) term, where \( 1 - \lambda > 0, \lambda \in (0, 1) \)
\[ \frac{1}{s^{1-\lambda}} = T_s \frac{1 + z^{-1}}{2} \left( \frac{1}{T_s} \right)^{1-\lambda} N_s(z^{-1}) D_s(z^{-1}). \]

After expanding (20), the recurrence formula of the fractional order integral component is obtained as
\[ u_i(k) = \left( \frac{1}{n_{es}} \right) \cdot \beta \left( \frac{1}{z} \right) \cdot \frac{1}{n_{ue}} e_i(k) - \frac{1}{n_{ue}} u_i(k), \]
\[ e_i(k) = n_{es} e(k) + (n_{ue} + n_{i1}) e(k-1) + (n_{ue} + n_{i2}) e(k-2) + + \ldots + (n_{ue} + n_{i1}) e(k-n_{ue} + n_{ue}) e(k-n_{ue}) e(k-n_{ue} + n_{ue}) e(k-n_{ue}), \]
\[ \beta \left( \frac{1}{z} \right) = (d_1 - d_0) u_i(k-1) + (d_2 - d_1) u_i(k-2) + + \ldots + (d_1 - d_0) u_i(k-n_{ue}) - d_0 u_i(k-n_{ue} + n_{ue}) e(k-n_{ue}), \]
where \( e(k-n_{ue}) \) is the previous \( n_{ue} \) error signal and \( u_i(k-n_{ue}) \) is the previous \( n_{ue} \) fractional order integral control action.

The fractional order derivative component is obtained by directly replacing \( s^\mu, \mu \in (0, 1) \) with any direct discrete-time mapper
\[ u_d(k) = \left( \frac{1}{n_{ed}} \right) \cdot \beta \left( \frac{1}{z} \right) \cdot \frac{1}{n_{ude}} e_d(k) - \frac{1}{n_{ude}} u_d(k), \]
\[ e_d(k) = n_{ed} e(k) + n_{ude} e(k-1) + n_{ude} e(k-2) + + \ldots + n_{ude} e(k-n_{ude}), \]
\[ \beta \left( \frac{1}{z} \right) = d_1 u_d(k-1) + d_2 u_d(k-2) + + d_1 u_d(k-n_{ude}). \]

Implementation guidelines

In an event-based control implementation, there are also some parameter choices that need to be done, apart from determining the direct discrete-time transfer function of the fractional order operators. The parameters that need to be customized depending upon the particularities of the process to be controlled are the nominal sampling time \( h_{nom} \), the maximum allowed time between two consecutive events \( h_{max} \) and the error variation threshold \( \Delta \). The other parameters of the event-based implementation are purely implementation variables, independent of the process. It is imperative that the parameters are used correctly inside the event detector and the control input generator for an efficient implementation. For an easy and quick reference, Table 1 provides a centralization of the parameters as well as some brief clarifications.

\[ \beta \text{ and } \gamma \text{ are weighting factors denoting the importance of the setpoint value in computing the error amount used to compute the fractional order proportional and derivative actions, respectively. The terms belong to the } (0, 1) \text{ interval. If there are no error weighting requirements of the fractional order control strategy or a simple implementation is desired, both terms should be chosen } \gamma = 1. \]

The nominal sampling time choice \( h_{nom} \) is identical to the sampling time in any discrete-time implementation. Shannon’s theorem should be respected.

The maximum allowed time between events is denoted through \( h_{max} \), which is a multiple of \( h_{nom} \). This is effective in implementing the safety condition. Choosing \( h_{nom} = h_{max} \) an event-based implementation leads to a pure discrete-time effect, eliminating all the benefits brought by the event-based approach. Hence, \( h_{max} \) should always be greater than \( h_{nom} \). The greater the value, the better the control signal computation is optimized, especially in the steady state regime. However, special attention should be given to choice of this parameter such that the safety condition is not completely eliminated. As a starting point, one should base the \( h_{max} \) choice on the dominant time constants of the process. Trial and error is also a viable option, as well as this possibility exists.

The previous error value is denoted by \( e_c \), in the available event-based literature [62]. Note that this error cannot include any weighting factor, in comparison to the error values used to compute the fractional order proportional and derivative gains respectively. The \( e_c \) variable stores the error at the previous event. Note that this value should be updated for every iteration of the control input generator. The value of \( e_c \) is used inside the event detector to check if an event should be triggered due to the error variation. The condition is usually written as \( (e(k) - e_c) \in [-\Delta_e, \Delta_e] \). The \([\Delta_e, \Delta_e] \) interval is composed by a predefined maximum error threshold. As a guideline, \( \Delta_e \) should be chosen as a small percentage, usually between 1–5% to the desired setpoint value.

The parameter \( h_{act} \) is a variable that stores the actual time elapsed since the previous event. The value should be updated every \( h_{nom} \) seconds whether an event is triggered or not. The formula that increments \( h_{act} \) can be written as \( h_{act} = h_{act} + h_{nom} \), making \( h_{act} \) another multiple of \( h_{nom} \). Note that \( h_{act} \) should be reset to 0 at the triggering of an event. The purpose of \( h_{act} \) is to determine if the safety condition is met.

### Numerical examples

The proposed event-based fractional order strategy is validated using numerical simulations targeting a wide range of processes from the simple first order plus dead time to a complex fractional order transfer function. The strategy is validated using fractional order PID controller case study. The case studies are intended to show the fractional order control versatility of the proposed methodology as well as to analyze its efficacy for a class of relevant processes.

All the controllers used in the numerical examples have been previously tuned using different fractional order methodologies.
The tuning procedure of each controller is irrelevant in proving the efficacy of the event based fractional order implementation. Both the Muir and CFE discrete-time approximations are tested with identical event-based implementation parameters in order to show the validity of any direct mapper for the present strategy and to assess the better discrete-time mapper choice for every example. Three implementation scenarios are performed and analyzed for every numerical example. The first test case analyzes the performance of the closed loop system obtained with the fractional order controller for a pure discrete-time implementation with the Muir recurrence-based direct discrete-time mapper from (7) and the CFE direct discrete-time mapper from (10). The test scenario validates the discrete-time fractional order controller on the selected process. The second and third test cases validate the proposed event-based fractional order control strategy, focusing on comparing the pure discrete-time realization of the fractional order controller to its event-based implementation.

First order plus dead time

For the first order plus dead time (FOPDT) model from

\[ H(s) = \frac{1}{4s + 1}e^{-s} \]  

(23)
a fractional order PI controller is computed by imposing a set of frequency domain specifications targeting the gain crossover frequency \( \omega_{gc} = 0.2 \text{ rad/s} \), the phase margin \( \phi_{pm} = 44^\circ \) and a certain degree of robustness to gain variations. The controller that meets these frequency domain constraints is

\[ C(s) = 0.0021 \left( 1 + \frac{162.6923}{s^{0.833}} \right) \]  

(24)
The Muir and CFE 5th order discrete-time mappers from (9) and (11) are used to compute the discrete-time approximation of the fractional order PI controller. The same sampling time \( T_s = 0.001 \) is used for both discretizations. The pure discrete-time implementation of the closed loop with every discrete-time controller is presented in Fig. 3a. The Muir and CFE approximations lead to a similar closed-loop system response.

The event-based fractional order PI implementation based on the Muir scheme is shown in Fig. 3b, while the CFE approximation is given in Fig. 3c. The implementation parameters are \( b = c = 1 \) for simplicity. The nominal sampling time is chosen such as \( h_{nom} = T_s = 0.001 \text{ s} \) and the safety condition time as \( h_{max} = 0.02 \text{ s} \), 20 times larger than \( h_{nom} \), while the error threshold is \( \Delta_e = 0.0005 \). The parameters are identical for both the Muir and CFE implementations. The simulations clearly show the cause of the control signal computations. It can be seen that in the transient response, the error triggered control prevails, whereas the safety condition is employed mostly in the steady-state regime. Both event-based fractional order implementations can be successfully used on the FOPDT process. However, during the 70 s duration of the tests, the Muir based controller computes the control signal 5824 times, the CFE controller 6508 times, improving the discrete-time control strategy which computes the control signal 70000 times.

Second order transfer function

The second order transfer function (SOTF) chosen as the second numerical example

\[ H(s) = \frac{78.35}{s^2 + 2.221s + 822} \]  

(25)
represents the dynamics of a smart beam. For the SOTF process a fractional order PID controller is tuned in [66] using an experimental optimization procedure. The obtained controller is

\[ C(s) = 0.0288 \left( 1 + \frac{3.4722}{s^{0.822}} + 28.743s^{0.822} \right) \]  

(26)

Fig. 4a shows the impulse response of the SOTF process with the classical discrete-time implementation of the fractional order PID controller with \( T_s = 0.001 \) s. The dynamics of the output variable are similar for both approaches. Fig. 4b shows the event-based fractional order PID controller discretized using the Muir recursion formula. The output of the closed loop system with the event-based controller is similar to the non-event-based Muir controller. The implementation parame-
ters are chosen as follows: $\beta = \gamma = 1$, $h_{\text{nom}} = T_1 = 0.001$ s, $h_{\text{max}} = 0.005$ s - 5 times greater than $h_{\text{nom}}$ and $\Delta_c = 10^{-4}$. The error threshold is chosen based on the amplitude of the closed-loop system response from Fig. 4a. The CFE event-based implementation is depicted in Fig. 4c. The process stabilizes with the same settling time as the Muir scenario. However, it can be observed that the amplitude of the output is 10 times larger using the CFE approximation, for the same event-based implementation parameters as in the Muir case.

From the control effort perspective, the Muir mapper computes the control signal value 368 times, the CFE controller 494 times, while the classical discrete-time controller command is evaluated 700 times for the 0.7 s in which the tests were performed. It can be stated that for this numerical example, the Muir based controller is the better choice both from the performance and the control optimization criteria.

**Second order plus time delay**

A Vertical Take-Off and Landing Platform is modeled through the second order plus time delay (SOPDT) transfer function

$$H(s) = \frac{22.24}{s^2 + 0.6934s + 5.244}e^{-0.8s}.$$  \hspace{1cm} (27)

The same tuning methodology as for the FOPDT numerical example is employed which tunes a robust fractional order PI controller that offers an open-loop system with a gain crossover frequency $\omega_{gc} = 0.4$ rad/s and a phase margin $\phi_m = 75^\circ$

$$C(s) = 0.0422 \left( 1 + \frac{492.2867}{s^{0.9288}} \right).$$  \hspace{1cm} (28)

Additional details regarding the real-life process and the control strategy can be found in [67].

Fig. 5a shows the closed-loop system response with the discrete-time Muir mapper and the CFE direct approximation using the sampling time $T_1 = 0.005$ s. It can be seen that the dynamics of the closed-loop system are similar for the classical implementation of the controllers. The closed-loop response of the event-based fractional order controller are $\beta = \gamma = 1$ for simplicity, $h_{\text{nom}} = T_1 = 0.005$ s, $h_{\text{max}} = 0.2$ s - 20 times larger than the nominal value and $\Delta_c = 0.01$, representing 1% of the unit step reference value.

The same tuning methodology as for the SOPDT experimental platform involves the experimental endorsement of the fractional order PI controller for reference tracking, disturbance rejection and robustness. In order to validate the proposed Muir event-based control strategy for the real-life test scenarios, disturbance rejection and robustness assessment are also taken into consideration.

Fig. 7 presents the closed loop response of the experimental platform equipped with the event-based fractional order PI controller. A 0.2 disturbance, representing 20% from the steady state performance, is added to the system at time 0.6 s.

---

Fig. 4. A fractional order PID controller applied to a SOTF process.
value, is introduced at moment $t = 20$ s. The event-based implementation parameters $b = c = 1$, $h_{nom} = T_s = 0.005$ s, $h_{max} = 0.2$ s and $\Delta_e = 0.01$ are the same as the ones used in Fig. 5. It can be seen that the output converges to the desired set point value, rejecting the disturbance. The event detector generates mostly process output events during the transient regime, which is to be expected from the event based implementation.

Fig. 8 targets the validation of the event based fractional order controller in a robustness test scenario. The fractional order controller from (28) has been tuned in [67] with respect to the iso-damping condition, ensuring a certain degree of robustness of the closed loop system to gain variations. Hence, the gain of the SOPDT process from (27) is altered by 30% and the response of the closed loop system to a unit step reference is analyzed. The test is identical to the one used to generate Fig. 5b, featuring the same event based implementation parameters. Comparing the nominal and the altered process responses, it can be easily observed that the settling time is increased from 6 to 10 s, as well as the overshoot soaring from approx. 10% to 40%. However, the event-based fractional order PI controller has a robust behavior, despite the large process' gain modification.
The tests performed in Figs. 7 and 8 show that the event based fractional order control strategy keeps the closed loop characteristics obtained through the discrete time implementation, such as disturbance rejection capabilities and robustness to gain variations. It is important to emphasize that an event based fractional order implementation does not aid in introducing a robust character of the closed loop system, being truthful to the already existing features of the controller implemented in the control input generator.

**Fractional order transfer function**

The last case study is focused on proving the applicability of the proposed event-based strategy to complex fractional order (FOTF) models. The transfer function

\[ H(s) = \frac{0.1}{s(0.005682^{1.7263} + 0.110315^{0.8882} + 1)} \]  

(29)

describes the braking effect of a non-Newtonian fluid, such as blood, on a transiting submersible. The main applicability of the study from [68], where the model and control have been developed, is in targeted drug delivery biomedical applications. One of the main request of the physical submersible is energy efficiency. This particular example is highly relevant to the proposed event-based fractional order methodology in order to show that the method can be useful to reduce computational effort in complex applications where energy efficiency is paramount. The control of the submersible’s position is done using the fractional order PD controller

\[ C(s) = 65.0028(1 + 0.0305^{0.6524}) \]  

(30)

**Fig. 9a** shows the response of the system with the classical discrete-time implementations using 5th order Muir compared to the CFE direct mapper with \( T_s = 0.001 \) s. In order to be true to the real process, the control signal that represents a PWM duty ratio is saturated between \([0, 1]\). Hence, the event-based fractional order control strategy is tested on another real life situation where the control signal range is limited. It can be seen that the dynamics of the closed-loop system are similar for the three discrete-time implementations.

The implementation parameters for the event-based Muir and CFE controllers are \( \beta = \gamma = 1, h_{\text{nom}} = T_s = 0.001 \) s, \( h_{\text{max}} = T_s = 0.005 \) s - 5 times greater than \( h_{\text{nom}} \) and \( \Delta_r = 0.01 \) representing 1% of the desired reference value. The closed-loop system responses using the event-based fractional order controllers with the Muir and CFE mappers are shown in Fig. 9b and c, respectively. Both controllers obtain similar responses as their non-event-based implementation. However, it can be clearly observed that the Muir realization of the discrete-time fractional order PD controller is the better choice.

From the control effort perspective, the Muir approximation computed the command value 380 times, whereas the CFE approach computed it 473 times. Both implementations reduce the 1500 number of computations realized with the non-event-based controller, bringing an improvement of more than 70% to the control effort.

**Conclusion**

The paper presents the theoretical background and the validation of a novel control strategy that combines the benefits of complex fractional order controllers with event-based implementations. The method is based on direct discrete-time mappers of fractional order operators. The theoretical aspects are presented for the fractional order PID controller, but any fractional order approach can be adapted given the provided guidelines. The study is validated using four numerical examples targeting various types of plants and all the fractional order PID controller variations.

The methodology has been successfully validated on every case study using various direct discrete-time mappers, showing the implementation versatility of the entire concept. For all four case studies, both the Muir and CFE direct mappers obtained a similar closed-loop performance as classical discrete-time implementations with a reduction of the control effort by at least 70% in every scenario. For all the chosen test cases, the Muir recursion based approximation provided a better closed-loop system response than the CFE approach.

The proposed methodology is most beneficial in industrial settings, where resource optimization and energy efficiency can drastically reduce the costs of various plants, as well as component wear over time.
Declaration of Competing Interest

The authors have declared no conflict of interest.

Compliance with Ethics Requirements

This article does not contain any studies with human or animal subjects.

Acknowledgment

This work was supported by the Research Foundation Flanders (FWO) under Grant 1S04719N and by the Romanian National Authority for Scientific Research and Innovation, CNCS/CCCDI-UEFISCDI, project number PN-III-P1-1.1-TE-2016–1396, TE 65/2018.

Event-based fractional order control implementation

Algorithm 1. The event detector

Result: The control signal $u$

Data: $k_p$, $k_i$, $k_d$, $\lambda$, $\mu$, $h_{nom}$, $h_{act}$, $\Delta e$, $Y_{sp}$

$h_{act} = 0$

$e_s = Y_{sp}$

while simulation ongoing do

process value $Y$

$e = Y_{sp} - Y$

if $(|e - e_s| > \Delta e$ or $h_{act} > h_{max})$ then

$u = \text{control_input_generator}(Y_{sp}, Y)$

$h_{act} = 0$

$e_s = e$

end

Algorithm 2. The fractional order control input generator

Result: The control signal $u$

Data: $ni_0, ni_1, \ldots, ni_n, di_0, di_1, \ldots, di_n, nd_0, nd_1, \ldots, nd_n, dd_0, dd_1, \ldots, di_n$

$e = Y_{sp} - Y$

$e_\beta = \beta Y_{sp} - Y$

$e_\gamma = \gamma Y_{sp} - Y$

/* compute the fractional order proportional component */

$u_{fp} = k_p * e$

/* compute the fractional order integral component */

$a = k_p * k_i * \text{pow}(2/h_{act}, -\lambda)$

$u_{fi} =$

$1/d_0 * (a * (ni_0 * e + (ni_0 + ni_1) * e_i_1 + \ldots + (ni_{n-1} + ni_n) * e_i_n + ni_n * e_i_{n+1}) - ((di_1 - di_0) * u_i_1 + (di_2 - di_1) * u_i_2 + \ldots + (di_n - di_{n-1}) * u_{i_{n-1}} + di_n * u_{i_n} + ud_{i_1} = e_i_{n+1} = e_i_n; e_i_n = e_i_{n-1}; \ldots; e_i_1 = e$

$u_i_{n+1} = u_i_n; u_i_n = u_i_{n-1}; \ldots; u_i_1 = u_{fi}$

/* compute the fractional order derivative component */

$b = k_p * k_d * \text{pow}(2/h_{act}, \mu)$

$u_{fd} = 1/dd_0 * (b * (nd_0 * e_\gamma + nd_1 * e_d_1 + \ldots + nd_n * e_d_n) - (dd_1 * u_d_1 + dd_2 * u_d_2 + \ldots + dd_n * u_d_n))$

$u_d_n = u_d_{n-1}; \ldots u_d_1 = e_\gamma$

$u_d_n = u_d_{n-1}; \ldots u_d_1 = u_{fd}$

/* compute the entire fractional order control signal */

$u = u_{fp} + u_{fi} + u_{fd}$
[62] Liu Q, Wang Z, He X, Zhou DH. A survey of event-based strategies on control and estimation 2014. doi: https://doi.org/10.1080/21642583.2014.880387.

[63] Hadian M, Aarabi A, Makvand A, Mehrshadian M. A new event-based pi controller using evolutionary algorithms. J Control, Autom Electr Syst 30 (29 2019). https://doi.org/10.1007/s40313-019-00519-1.

[64] Liu R, Wu J, Wang D. Event-based reference tracking control of discrete-time nonlinear systems via delta operator method. J Franklin Inst 2019;356(17): 10514–531, special Issue on Distributed Event-Triggered Control, Estimation, and Optimization. https://doi.org/10.1016/j.jfranklin.2018.07.022.

[65] Ruiz A, Beschi M, Visioli A, Dormido S, Jiménez J. A unified event-based control approach for foptd and iptd processes based on the filtered smith predictor. J Franklin Inst 2016;354. https://doi.org/10.1016/j.jfranklin.2016.11.017.

[66] Birs I, Folea S, Prodan O, Dulf E, Muresan C. An experimental tuning approach of fractional order controllers in the frequency domain, Vol. 10: 2020. https://doi.org/10.3390/app10072379.

[67] Birs I, Muresan C, Nascu I, Folea S, Ionescu C. Experimental results of fractional order PI controller designed for second order plus dead time (SOPDT) processes. In: 2018 15th International Conference on Control, Automation, Robotics and Vision, ICARCV 2018. doi: https://doi.org/10.1109/ICARCV.2018.8581241.

[68] Birs I, Muresan C, Copot D, Nascu I, Ionescu C. Identification for control of suspended objects in non-newtonian fluids. Fract Calc Appl Anal 2020. doi: https://doi.org/10.1515/fca-2019-0072.