Preprints, 4th IFAC Conference on
Advances in Proportional-Integral-Derivative Control
Almería, Spain | June 12-14, 2024

Teaching PID to Future Professionals as a Contribution to Fill a Historical Gap

Alberto Leva *

* *DEIB, Politecnico di Milano, Italy (e-mail: alberto.leva@polimi.it).*

Abstract Several authoritative literature sources evidence a gap between how control is taught in the academia and how it is applied in the engineering practice. This paper argues that an important obstacle toward filling the said gap resides in the way basic control blocks like PIDs are treated in most control curricula, especially at the fundamental (typically, BSc) level. More specifically, the problem is that too little focus is set on some functionalities that not only are necessary for the correct operation of those blocks, but are also required to reach a firm grasp on how a control scheme of professionally realistic complexity is structured. Based on the considerations just sketched, a didactic proposal to embrace such functionalities right from the basics of PID teaching is then formulated.

Keywords: PID control, Control education, Control engineering, Industrial control.

1. INTRODUCTION AND RESEARCH QUESTIONS

The existence of a gap between academic teaching and professional practice in control engineering was observed several times in the literature, and the forecast by Shinskey (2002) that "the gap is not closing" seems to still find support in subsequent works, see for example those by Silverstein et al. (2016) and Bequette (2019).

Here we focus on a seldom treated but relevant aspect of the gap, namely an under-emphasis on functionalities of basic control blocks beyond their LTI description — a matter not exclusive to the PID, but most likely best taught on the PID. We show that a shallow knowledge of this subject has at least two consequences. One is to hinder the comprehension of how a modulating loop is operated; the other – maybe less evident but in the opinion of the author at least equally detrimental – is to make it difficult for the student to understand (and in perspective design) articulated control schemes, where the coordination of the above functionalities, as well as the logic that this coordination entails, often plays a crucial role. Such schemes are very frequently encountered in process control — whence maybe the nature of most literature works that evidence the addressed gap; however, a firm understanding (hence a solid managing capability) of "large" projects is becoming a necessity for the control engineer, almost independently of the domain where he/she has to operate. This said, our research questions are posed below.

- Q1 Does (PID-centred) control education devote too few space to the above "additional functionalities", detailed in the following, and does this result in a relevant competence gap?
- Q2 If so, is PID teaching fit – and in the affirmative case, with which enrichments – to help fill the said gap?

In the rest of this paper we carry out a state of the art analysis in the light of these questions, support it with literature evidence up to an extent compatible with the available space, and formulate some guidelines for extending the common teaching practice. Of course the proposal cannot fill the gap by itself, yet it can serve as an enabler to address several of the problem facets evidenced in the analysis. In accordance with the proposal some material is also being prepared, within a long-term project preliminarily described in (Leva, 2023) and that will be further treated in subsequent works.

2. RELATED WORK AND PROBLEM

We start our answer by listing – with no exhaustiveness claim – the principal functionalities we are talking about, and that in the following we are calling "extras" as additions to the mere LTI control law. We give here just the bare essential information, the brief examples reported later on will provide some additional detail.

Two degree of freedom (2-dof) realisation. This is in fact still LTI, but allows separating set point following and disturbance rejection.

Antiwindup. This requires no explanation — if not for suggesting to not call the counteracted phenomenon *integral* windup. A controller with no integral action but a large enough gain and a large enough dominant time constant can produce the very same effect.

Increment/decrement locks. These are boolean inputs that, when true, prevent the control signal to increase or decrease. They are useful when control saturation values are condition-dependent, most typically in the outer loops of cascade controls.

Rate limits. Managing these is analogous to antiwindup, just acting not on the control signal but on its derivative with time.

Incremental vs. positional form. The former refers to computing at each step the *variation* f the control signal, and then accumulating; the latter is the opposite. This

matter has a sometimes tricky interplay with antiwindup, and rate limits management if present.

Biasing. This refers to an additional input that acts additively on the control signal. The addition must take place upstream the saturation management mechanism, whence here too an interplay with antiwindup and rate limits. Biasing is useful for compensation, feedback/feedforward and decoupling schemes.

Tracking. To avoid confusion this is not about set point following, but rather about the presence of a boolean input, typically named TS for Track Switch, that when true constrains the control signal to follow a numeric input, typically called TR for Track Reference. Tracking is useful for controller switching and the realisation of automatic/manual mode.

Quantisation. In some cases not accounting properly for the input and output resolution (most often dictated by analogue/digital/analogue converters) can result in undesired limit cycles; in very low-end arithmetic platforms, precision can even impact the controller singularities up to significantly altering the behaviour of a loop — and most notably, managing the matter imposes a *lower* limit to the sampling time.

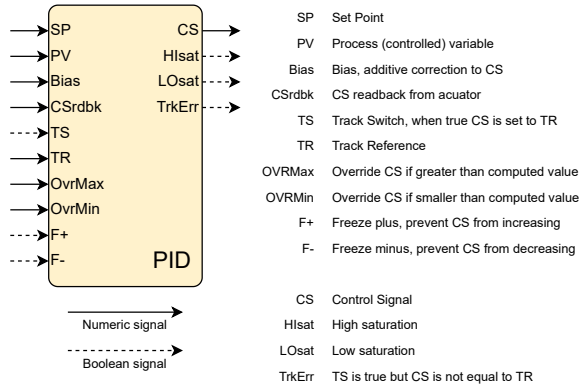

Figure 1. A PID block as typically seen in an industrial control development tool.

For space reasons we cannot treat all the extras in this paper, others will follow. Suffice however to notice that as a result of them, the way a PID is seen in a tool for industrial control development (exemplified in Figure 1) is quite different from the way it is – and must be – initially introduced in a class.The students must be led to relate those two descriptions and to understand (simplifying for brevity) that the latter is for parameter tuning, but for a proper loop management a controller model needs to embrace the former. And as a consequence, they should undertand that the "extras" are not "implementaiton facts that the theory does not address"; on the contrary, they need treating *methodologically.*

With the list above in mind, let us now throw a high-level glance at the literature. Table 1 shows the results of some Google Scholar search queries carried out at the time of this writing. The queries were for "PID control" and "teaching" plus several other terms, that for the purpose of this research were categorised as follows.

- *Implementation*, referring somehow to the way the controller is realised.
- *Domain*, referring to some particular field in which the controller is applied.
- *Structures*, referring to some particular scheme to which the controller belongs.
- *Technology*, referring to the hardware/software architecture where the controller is deployed.
- *Extras*, referring to the functionalities listed above.

For each term the table reports the number of found references; the rows are ordered by this number, decreasing. No doubt the search outcome must be taken as a definitely coarse representative of the reality. Some categories may be seen as partially intertwined, and some terms might need refining (for example, "track switch" was chosen because "tracking" would catch all the references to *set point* tracking, which is not the intended result). However, deferring a finer analysis to future woks, a couple of interesting remarks can already be made.

The first one is the wide magnitude span, as numbers range from some thousands down to a few units — and the author doubts that introducing the refinements sketched above would change things significantly. The second and most important one is the (quasi-)diagonal aspect of the resulting hits/categories matrix. Even assuming that "digital" and "real time" can occur in combination with other terms, which would emphasise the relative importance of those two terms with respect to the others, it is quite evident that the literature establishes sort of a ranking from "implementation" down to "extras". The only "off-diagonal" items are "compensation", that however has several meanings, "ratio control", that is in fact less frequent than the other mentioned structures, and finally "2-dof" and "antiwindup", that among the considered extras are most likely the least "extra", particularly when viewed in a teaching context.

We deem the remarks above sufficient to say that extras receive less attention than other subjects, and a more detailed analysis of individual papers on PID teaching confirms the statement. When professional (but not control-exclusive) development tools such as MATLAB/Simulink or LabVIEW are used, the treatise stays mostly in the LTI domain (Keller, 2000; Yao et al., 2009; Huba and Simunek, 2007; Tran et al., 2019). Antiwindup is often mentioned but with no discussion on the used method and its implications, and there is hardly any evidence of the other extras above. The same applies to experiences with open tools such as Scilab/Scicos, see e.g. Tona (2006). When industrial control equipment is used, this most frequently happens on a single-loop basis (Vargas et al., 2023), and the same applies when more "open" systems – such as the Arduino – come into play to teach how PID software is written (Moura Oliveira and Hedengren, 2019); also in such cases, the mentioned extras are seldom mentioned. In some cases the didactic scope trespasses the single loop to embrace the control of a process, see for example the work by Warke (2012), but still the additional functionalities of modulating blocks and the associated logic play a quite marginal role.

The above said, to answer Q1 we now need to discriminate whether or not this reduced focus results in a relevant competence gap. Here the author would be tempted to answer yes based on his experience. No doubt the assessment

Table 1. Categorised results from Google Scholar having as search terms "PID control" and "teaching" and those indicated in each row; "∗" stands for "any string".

Search term	No. of hits	Implementation	Domain	Structures	Technology	Extras
digital	6880	✓				
real time	6870	✓				
robot∗	4710		✓			
manufacturing	3640		✓			
process control	3340		✓			
compensation	3070			✓		
mechatronics	2900		✓			
feedforward	1850			✓		
cascade	1640			✓		
batch	1280			✓		
decoupling	1070			✓		
2-dof	705					✓
antiwindup	534					✓
PLC	440				✓	
IEC	411				✓	
DCS	382				✓	
ratio control	210			✓		
locks	152					✓
rate limit∗	63					✓
incremental form	45					✓
bias input	11					✓
track switch	5					✓
quantiz∗	3					✓

of individual loops is the foundation of any good control maintenance process (Bauer et al., 2016, 2019), but along the years the author has seen numerous control malfunctions just caused by the absence of a lock, or some block not being forced to tracking when it should have been, or similar flaws. For the sake of completeness, however, an authoritative support for the statement under question can be found by going through the book by Wang (2020), that devotes a significant amount of space to antiwindup techniques, the effects of quantisation, and the use of PIDs in control structures. The point is that the said work is not a textbook, nor (which is not a criticism) does it strongly focus on industrial development systems. In fact, notions about the functionalities we talk about are scattered amidst a heterogeneous technical or even product-specific literature, and often learnt in the field from the experience of practitioners — though it was noticed several years ago (Li et al., 2006) that in the PID domain " development focuses on providing additional and supervisory features, including support for various controller structures". On the university education side attempts were made to fill the academia-practice competence gap by having industry people participate into teaching, see e.g. Hoernicke et al. (2017), and this certainly helps the student contextualise the learnt notions into an application environment, but it still seems that the functionalities we address are seen as "implementation accidentals" instead of an integral part in the design of a complete control strategy (Maggio and Leva, 2011; Leva and Cimino, 2019). A high-level view on the matter, complementary to the very development-centric attitude of the papers just quoted and supporting the advantages of addressing all the mentioned functionalities and extras in a unitary manner, can be found in the strongly industry-geared paper by Brisk (2005) — needless to notice, not an education-oriented work either.

3. A DIDACTIC PROPOSAL

We come now to formulate a proposal – or maybe better, the proposal of an integration to common teaching practice – to address the evidenced problem. After "standard" PID teaching, the suggestion is to guide the class through a reasoning path made of the following steps:

(1) the control signal has to abide by physical limits, inevitably in magnitude and sometimes also in rate;
(2) sometimes the control signal needs to depend (also) on other inputs than the error;
(3) sometimes it may need to not even depend at all on the error;
(4) in any case, the state of the controller must be consistent with its input(s) and its output;
(5) as such, besides computing the control signal, a controller must contain functionalities to handle the above
(6) and besides additional inputs and outputs, realising the said functionalities requires the controller to have a Human machine Interface part, which may need to contain additional states.

We do not report here examples to help understanding the idea. There are plenty in any application domain, any instructor can find the most appropriate ones. The key points are however the last three, as experience indicates that the students easily recognise the importance of "keeping the state consistent" from a theoretical viewpoint, but find it difficult (i) to figure out how this can be realised, *first* in terms of a dynamic system and *then* as software, and (ii) to understand how an HMI should be built and operated.

Incidentally, it would be beneficial to expose the students to digital controller realisation as soon as possible, and this is why in the following we are reasoning in the discrete time domain. The author is aware of the entailed difficulties, but based on a 20+ years experience with basic control teaching to computer engineers the objective is well attainable if one just accepts to not delve into all the underlying mathematics. For example, one could refrain from treating the $\mathcal{Z}$ transform and just introduce as operators the one-step advance z and as a consequence the variation one $\Delta = 1 - z^{-1}$, coming as a consequence to the fixed-rate time derivative approximation Δ/T_s, where T_s is the sampling period.

The author also acknowledges the encountered objection that the subject above could seem inappropriate for a basic course, but based again on experience several control malfunctions are due to improper management of the addressed functionalities — as easily happens, for example, if a stateful HMI is created by "software only" people, with no notion of dynamics. It is now the time to spend a few words on how the idea could be contextualised in basic and more advanced courses.

3.1 Basic courses

For obvious reasons, basic courses are centred on the single loop. At this level one should concentrate on saturation management and auto/manual mode.

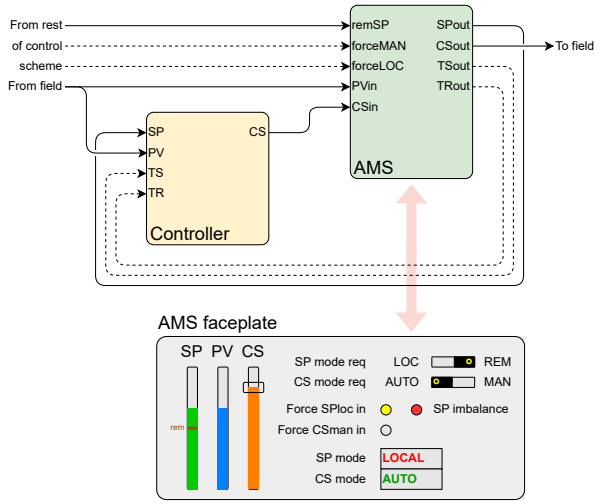

Figure 2. A (PID) controller connected to an AMS and its faceplate.

An effective way to address the subject is to introduce and discuss the Auto-Manual Station (AMS) component, viewing its HMI or "faceplate" as a dynamic system intertwined with the controller. Referring to Figure 2, for example, when the controller is in AUTO(matic) mode the CS (Control Signal) slider must track the computed control while in MAN(ual) mode the same slider mut act as a means for the operator to govern that control. We omit further details for space limitations, the reader cen refer to the online material announced in Leva (2023) and progressively made available.

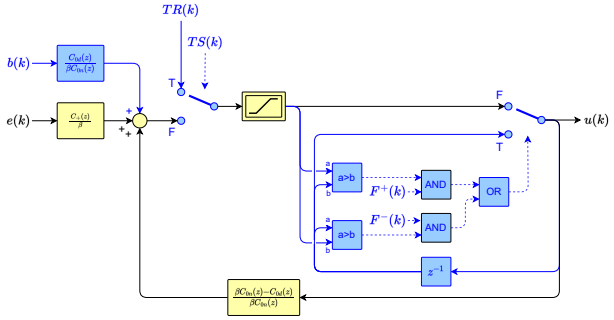

Figure 3. A discrete-time (PID) controller realised in internal feedback form with bias, tracking mode and increment/decrement locks.

It is also important that the addressed functionalities be explained in terms of dynamic systems and not directly as pieces of code, because the latter way would support the already mentioned idea that these are just implementation accidentals. For example, Figure 3 reports the diagram of a discrete-time LTI controller realised by means of internal positive feedback and endowed with bias $b(k)$, tracking mode and increment/decrement locks (logic inputs F^+ and F^-). The scheme is general, but setting

$$\begin{aligned} C^+(z) &= 1, \\ \beta &= \frac{T_i}{K(T_i+T_s)}, \\ C_{0n}(z) &= K\left(z(T_i+T_s)-T_i\right), \\ C_{0d}(z) &= T_i(z-1) \end{aligned} \tag{1}$$

results in a 1-dof PI with gain K and integral time T_i, discretised by the backward difference method at timestep T_s. The reader can recognise the "textbook" version of such a controller in the pale yellow blocks, including the saturation one, with (1) substituted for the contained entities; the blue blocks correspond to the additional functionalities.

Another important subject to address is the influence of saturation management on the loop transients undergone on exiting saturation. To this end it suffices to write the controller output equation as

$$u(k) = c_C x_C(k) + d_C e(k) \tag{2}$$

where $x_C(k)$ is the state vector, $e(k)$ the error, $u(k)$ the control signal, and in the SISO case c_C is a row vector and d_C a scalar. When saturation constrains $u(k)$ to equal a given limit $\overline{u}$, any $x_C(k)$ such that

$$c_C x_C(k) + d_C e(k) = \overline{u} \tag{3}$$

is consistent with the controller operating condition. However, when saturation ends, the value of x_C computed at the last step in saturation becomes part of the initial state of the closed loop, that from that step on comes back to evolving linearly. A few experiments are enough to convince the students that the differences in the resulting transients are not due to some "software bug" but simply inevitable, as computing the controller state in saturation is an inherently under-determined problem.

3.2 Advanced courses

Advanced courses have to look beyond the single loop, thereby addressing other "extras" in the sense meant herein. In this respect, the students should be helped understand that the problem is (almost) invariantly the same, that is, preventing the internal state of the controller from becoming inconsistent with its inputs and outputs.

The point here, in contrast to basic courses, is that the "controller" may not be just a single block, but rather a structure made of several blocks — where PID-type ones most often provide the backbone, incidentally. As a result, preserving the state consistency may entail inter-block communication — an interesting starting point for further teaching on a conscious use of standards for distributed systems like the IEC 61499, but we ar not addressing this matter herein. Expanding the list of Extras in Section 2 one can devise various activities: a few examples follow.

More on tracking. The controller-AMS connection required to introduce TS and TR at the level of the single controller block. Thinking of structures, the same signals serve for managing the switchover of different controllers to act on a single actuator: a switch block decides which control signal reaches the process, this signal is sent as TR to all controllers, and all of them but the one that is acting on the process have TS set to true. Another case is with cascade controls: if the inner controller is set to tracking (or manual) mode, the outer one must be forced into tracking as well because its loop is open. In this case one has also to decide which value should be presented as TR to the outer controller: a common choice is the measured value of the inner controlled variable, so that on exit from tracking the inner controller starts from a rest state, but depending on the problem other alternatives could be explored.

Biasing. When the output of a dynamic block is subjected to an additive correction, this must be done *upstream* any output saturation management mechanism, or the state of the block will be made consistent with the uncorrected output. This is the reason for endowing controllers with a bias input as seen from a system-theoretical standpoint. Thinking of control structures, two paradigmatic cases can be addressed. The first one, very simple, is feedforward compensation: on this no explanation is needed here.

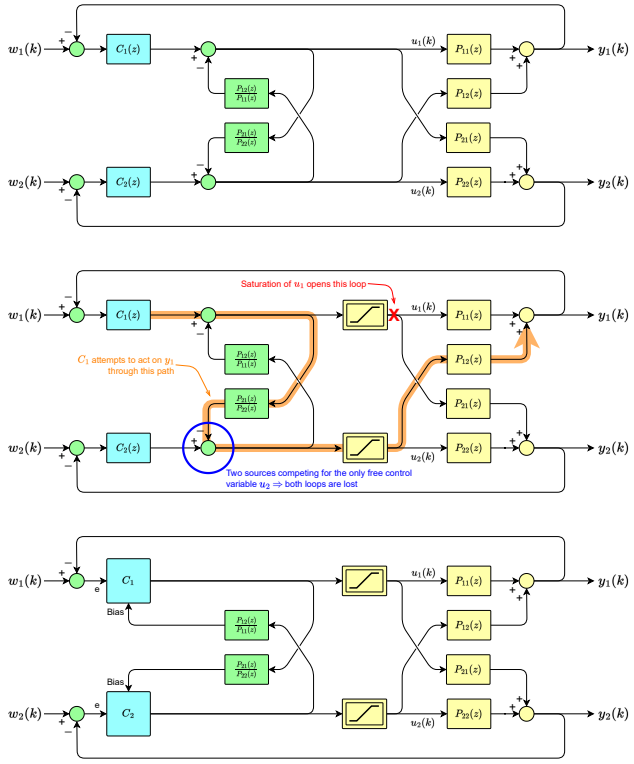

Figure 4. Role of the bias input of (PID) controllers in MIMO control via backward decoupling.

The second one, illustrated in Figure 4 in the 2×2 case without loss of generality, is more subtle and involves backward decoupling. The point is that the antiwindup of the employed controllers (C_1 and C_2 in the figure) acts on *their* outputs, which are not the signals subjected to saturation owing to the presence of the decoupler (green blocks). The scheme "as taught" is shown at the top of the figure, while at the centre the problem is illustrated: if one of the physical control signals hits a saturation limit this most likely does not apply to the output of the corresponding controller, which will try to influence its own controlled variable (in the figure, y_1) via the evidenced path. The result is the one available control signal taking orders from two masters, hence the loss of both loops.

Once clarified that the antiwindup mechanism of the individual controllers is made useless for the problem above owing to saturation not acting on their outputs, the scheme "as practiced" is shown at the bottom of Figure 4. By applying the decoupling corrections via the bias inputs of C_1 and C_2 – i.e., as said above, upstream their saturation management – the problem is solved.

3.3 Brief discussion

We have seen some examples of possible teaching subjects about the evidence "extras". For basic courses we omit further ones like the effects of positional vs. incremental forms for the actions of a PID in the presence of value or rate control saturation, but there is plenty of material to work with to convince the students that not only there is more to a controller than its LTI law, but most important that all of this *can be viewed and analysed as a dynamic system.* If the result just sketched is achieved, for a basic course in the author's opinion it is more than enough.

Also for advanced courses any instructor can easily find other examples and cases to address besides those shown. From a cultural viewpoint, the students should understand that sometimes the *structure* of a control system changes by physical events (e.g., a saturation limit) or operator commands (e.g., a loop forced to manual). On the one hand, this could be a starting point for studying switching systems, and most important, their control engineering relevance. On the other hand, the same starting point could lead to a system-theoretically conscious approach to the industry standards (like the mentioned IEC one) relative to the realisation of control applications.

In all the cases seen, despite the conveyed ideas are general with respect to the controller structure, no doubt the PID (or sometimes even just the PI) is fit, and is probably the simplest structure to address the required matter. From the author's standpoint this allows to answer Q2 affirmatively.

4. IMPLEMENTATION

In synthesis, the proposal is to teach PID with "industrially realistic" blocks as soon as possible, treating each "extra" the way Figure 4 suggests when observed from top to bottom. This means starting from an LTI description of the system to address and carry out the intended pedagogy. The second step is evidencing the problems neglected in the LTI-only context, characterising them as nonlinear and/or time-varying characters of the real system. The final step is to introduce and explain the addressed extra(s).

With the adaptations that each instructor can introduce, the teaching path to implement the proposal is quite clear.

Things are a bit more problematic as for the tools to use, as the proposal requires the possibility of working at the level of the control algorithm (in basic courses seeing and listening to explanations, in advanced ones also writing), of building/managing HMIs, and of carrying out both batch and interactive simulations in contexts that evidence the addressed problems and also allow to give them some physical meaning (i.e., not only with plants represented as transfer functions).

At present the author uses OpenModelica, but any other block-oriented tool would fit. The weak point of the typical Modelica tool is however the limited capability as for interactive simulation, and particularly the difficulty in creating and managing "industry-like" HMIs. This is done at the moment with LabVIEW, that in some sense has symmetric pros and cons with respect to Modelica. An integration of the two worlds is being sought, and can be created by exploiting the Functional Mockup Interface (FMI) standard (Blochwitz et al., 2011): the challenge is to do so in a way that requires on the part of the students an acceptable time expenditure at the startup of the didactic activity; at the time of this writing, the above is work in progress.

5. CONCLUSIONS AND FUTURE WORK

We argued that in PID teaching we should expose the students to "extra" controller functionalities (in the sense shown above) earlier than we typically do. There are undoubted difficulties in doing so, but a lack of culture on the matter has at least two consequences. First, professional experience indicates that an absent or improper management of such extras (tracking, biasing and so forth) is the cause of several malfunctions at bot the single loop and the control strategy level. Second, the students may retain the idea that the same extras are not part of the design engineering of a control system, but rather just implementation accidentals to be possibly addressed downstream, not within the said design.

We have provided a didactic proposal sketch to cure the evidenced issue in basic and advanced courses. Future work will be directed at refining the proposal, as well as to the identification of the most suited set of tools to put the devised ideas to work.

REFERENCES

Bauer, M., Auret, L., Bacci di Capaci, R., Horch, A., and Thornhill, N. (2019). Industrial PID control loop data repository and comparison of fault detection methods. *Industrial & Engineering Chemistry Research*, 58(26), 11430–11439.

Bauer, M., Horch, A., Xie, L., Jelali, M., and Thornhill, N. (2016). The current state of control loop performance monitoring–a survey of application in industry. *Journal of Process Control*, 38, 1–10.

Bequette, B. (2019). Process control practice and education: past, present and future. *Computers & Chemical Engineering*, 128, 538–556.

Blochwitz, T., Otter, M., Arnold, M., Bausch, C., Clauß, C., Elmqvist, H., Junghanns, A., Mauss, J., Monteiro, M., Neidhold, T., Neumerkel, D., Olsson, H., Peetz, J., and Wolfs, S. (2011). The Functional Mockup Interface for tool independent exchange of simulation models. In *Proc. 8th international Modelica conference*, 105–114. Dresden, Germany.

Brisk, M. (2005). Process control: potential benefits and wasted opportunities. *Australian Journal of Electrical and Electronics Engineering*, 2(1), 41–48.

Hoernicke, M., Horch, A., and Bauer, M. (2017). Industry contribution to control engineering education: an experience of teaching of undergraduate and postgraduate courses. *IFAC-PapersOnLine*, 50(2), 133–138.

Huba, M. and Simunek, M. (2007). Modular approach to teaching PID control. *IEEE Transactions on Industrial Electronics*, 54(6), 3112–3121.

Keller, J.P. (2000). Teaching PID and fuzzy controllers with LabVIEW. *International Journal of Engineering Education*, 16(3), 202–211.

Leva, A. (2023). An MSc-level course on ICT for control systems engineering. *IFAC PapersOnLine*, 56(2), 4681–4686.

Leva, A. and Cimino, C. (2019). Teaching to design control applications with coordinated modulating and logic functions. In *Proc. 18th European Control Conference*, 3047–3052. Naples, Italy.

Li, Y., Ang, K., and Chong, G. (2006). Patents, software, and hardware for PID control: an overview and analysis of the current art. *IEEE Control Systems Magazine*, 26(1), 42–54.

Maggio, M. and Leva, A. (2011). Teaching to write control code. *IFAC Proceedings Volumes*, 44(1), 7292–7297.

Moura Oliveira, P. and Hedengren, J. (2019). An APMonitor temperature lab PID control experiment for undergraduate students. In *Proc. 24th IEEE International Conference on Emerging Technologies and Factory Automation*, 790–797. Zaragoza, Spain.

Shinskey, F. (2002). Process control: as taught vs as practiced. *Industrial & Engineering Chemistry Research*, 41(16), 3745–3750.

Silverstein, D., Vigeant, M., and Staehle, M. (2016). How we teach process control: 2015 survey results. In *Proc. 123rd ASEE Annual Conference & Exposition*. New Orleans, LA, USA.

Tona, P. (2006). Teaching process control with Scilab and Scicos. In *Proc. 2006 American Control Conference*, 280–285. Minneapolis, MN, USA.

Tran, L., Sun, Y., Guan, R., Saeed, J., Wang, L., and Radcliffe, P. (2019). Development and outcomes of teaching PID control in classroom with hands on learning experience. In *Proc. 2019 IEEE International Conference on Industrial Technology*, 1041–1047. Melbourne, Australia.

Vargas, H., Heradio, R., Donoso, M., and Farias, G. (2023). Teaching automation with Factory I/O under a competency-based curriculum. *Multimedia Tools and Applications*, 82(13), 19221–19246.

Wang, L. (2020). *PID control system design and automatic tuning using MATLAB/Simulink*. Wiley, Hoboken, NJ, USA.

Warke, N.V. (2012). LabVIEW-teaching aid for process control. *ACEEE Int. J. on Control System and Instrumentation*, 3(02).

Yao, J., Limberis, L., Williams, R., and Howard, E. (2009). An efficient pid control teaching module with LabVIEW simulation. *Computers in Education*, 19, 30–41.