A tuning procedure for the Cascade Proportional Integral Retarded Controller ⋆

Kevin López, **Sabine Mondié**, **Rubén Garrido**.

Departmento de Control Automático, CINVESTAV-IPN, Mexico City, 07360, Mexico (e-mail: klopez, smondie, garrido @ctrl.cinvestav.mx).

Abstract: This work proposes a tuning methodology for the Cascade Proportional Integral Retarded (CPIR) controller applied to a class of second order systems. Unlike the Proportional Integral Retarded (PIR) control law, the CPIR is composed of two nested loops, each of them having its own controller. The inner loop is regulated through an Integral Retarded (IR) Controller and the outer loop uses a proportional (P) controller. It is worth remarking that the stability of both loops must be simultaneously guaranteed, a problem not appearing in the PIR controller. The proposed tuning methodology is based on the spectral abscissa analysis of the Outer loop. The IR controller tuning rules and a double root placement analysis in the CPIR quasy-polynomial allows writing an analytical form of the P proportional gain. Subsequently, this gain is computed by solving a transcendental nonlinear equation. The proposed tuning method is assessed by means of experiments in a laboratory prototype.

Keywords: Time-delay systems, delay-based control, retarded control, cascade proportional integral retarded control.

1. INTRODUCTION

Several delay-based algorithms have been proposed including the Proportional Retarded (PR) Suh and Bien (1979); Villafuerte et al. (2013) and the Proportional-Integral-Retarded (PIR) Chen (1987); Ramírez et al. (2016). It is worth noting that tuning of some of these controllers relies on the use of the spectral abscissa analysis methods. Moreover, these control laws are the counterparts of the standard Proportional Derivative (PD) and Proportional Integral Derivative (PID) controllers. Delay-based controllers exhibit measurement noise filtering properties making unnecesary the use of filters as it has been shown in Ramírez et al. (2015) for the Integral retarded (IR) controller applied to the velocity control of servodrives, which is traditionally performed by Proportional Integral (PI) controllers using filtered measurements. Two problems arise when introducing a time-delay in a feedback control law. The first difficulty is related to closed-loop stability, and it happens owing to the infinite number of roots generate by the time-delay. Another key issue is the tuning procedure for the above algorithms. Those problems are addressed in the case of the PR Villafuerte et al. (2013), IR Ramírez et al. (2015), and PIR Ramírez et al. (2016) controllers by imposing a dominant triple root in the closed-loop system that produces the maximum exponential decay rate. Moreover, these references give explicit tuning rules for computing the controllers gains as well as the time delay. Recently, a Cascade Proportional Integral Retarded (CPIR) algorithm has been proposed López et al. (2017). It corresponds to the time-delay version of the Cascade Proportional-Proportional Integral (P-PI) algorithm widely used in practice for controlling servodrives Ellis (2012). In the P-PI controller, a proportional controller closes an outer position loop, and an inner PI controller regulates a velocity loop The cascade nature of the P-PI controller offers several advantages. The velocity and position loops are easily tuned, and the servodrive may function in velocity mode by simply disconnecting the position loop. The CPIR controller mimics the above scheme by using an IR controller in the inner velocity loop while keeping the P controller in the outer loop. However, the fact the velocity and position loops of the CPIR controller may function independently complicates its tuning. Note that if the tuning of the IR controller renders the velocity loop stable, then, it does not imply a stable position loop. In the same way, tuning of the position loop does not guarantee a stable velocity loop. This problem has been tackled in López et al. (2017) by tuning the IR controller in the velocity loop using the tuning rules proposed in Ramírez et al. (2015), and subsequently tuning the P controller closing the position loop by numerically minimizing a performance index. The purpose of this work is to propose a new tuning procedure for the CPIR controller, which is based on the analytic formulae previously proposed in Ramírez et al. (2015) and Ramírez et al. (2016). Unlike the approach in López et al. (2017), the tuning procedure does nor rely on minimizing a numerical index. Instead, the dominant roots of the inner loop quasipolynomial, together with the tuning rules of the IR and the PIR controllers allow writing the outer loop quasipolynomial as a function of the position controller proportional gain. Subsequently, this gain is computed by solving a transcendental nonlinear equation. The outline of the work is as follows. After describing the servodrive model and the IR and PIR controllers, the proposed tuning procedure is described in detail. Later, an experimental

⋆ This work was supported by CONACyT-MEXICO under projects 222140 and 180725. The work of Kevin López was supported by a grant from CONACyT-MEXICO.

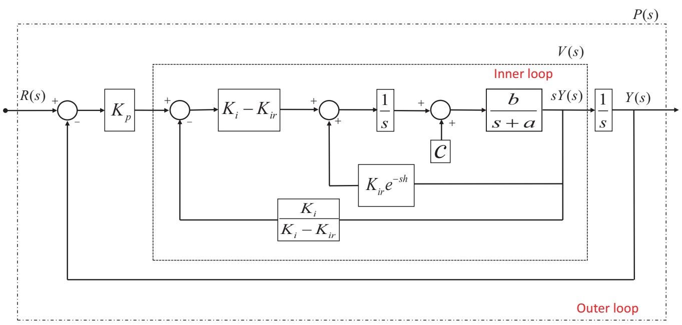

Fig. 1. CPIR block diagram.

study allows assesing the performance of a DC servodrive in closed loop with the CPIR controller tuned under the proposed procedure. The paper closes with some remarks and mentions the future work on the matter.

2. CPIR CONTROLLER TUNING

2.1 Servodrive model, IR and PIR controllers

Consider the dynamics of a servodrive

$$\ddot{y}(t) + a\dot{y}(t) = bu(t) + c \tag{1}$$

with $a > 0$, $b \neq 0$ and $c > 0$ a constant disturbance, in closed loop with the CPIR controller depicted in Fig.1. The IR controller regulates the inner loop fed with velocity measurements and has three parameters to tune, i.e. the gains K_i and K_{ir}, and the time-delay h. The outer loop is closed through a P controller with gain K_p and is fed with position measurements. The equation describing the CPIR controller is

$$\dot{u}(t) = (K_i - K_{ir})K_p e(t) - K_{ir}\dot{y}(t-h) + K_i\dot{y}(t) \tag{2}$$

Define the position error as $e(t) = r - y(t)$ with r a constant desired position. Taking the time derivative of $e(t)$ and using (1) produces the next error dynamics

$$\ddot{e}(t) = -a\dot{e}(t) - bu(t) - c \tag{3}$$

Now, define

$$z(t) = -bu(t) - c \tag{4}$$

Introducing the above definition into (3) produces

$$\ddot{e}(t) = -a\dot{e}(t) + z(t) \tag{5}$$

The time derivative of (4) considering control law (2) in terms of position error is

$$\dot{z}(t) = -b(K_i - K_{ir})K_p e(t) + bK_{ir}\dot{e}(t-h) - bK_i\dot{e}(t) \tag{6}$$

Therefore, equations (5) and (6) define the position loop dynamics related to the CPIR controller. Its stability is defined by the roots of the next characteristic quasipolynomial

$$P(s) = s^3 + as^2 + sb(K_i - K_{ir}e^{-sh}) + bK_p(K_i - K_{ir}) \tag{7}$$

It is also possible to obtain from Fig.1 the velocity loop quasipolynomial associated to the IR controller

$$V(s) = s^2 + as + b(K_i - K_{ir}e^{-sh}) \tag{8}$$

2.2 The tuning method

At this point, it is worth recalling the tuning rules for the IR controller proposed in Ramírez et al. (2015)

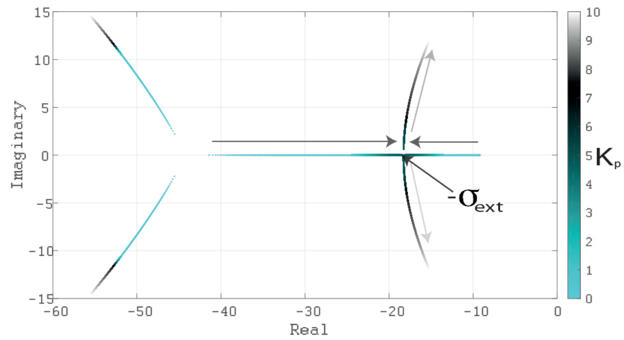

Fig. 2. Behavior of the roots of quasipolynomial (14) for $K_p \in (0, 10]$ and and $\sigma_{int} = 34.9015$.

$$h = \frac{1}{\sigma_{int}}, \tag{9}$$

$$K_i = \frac{\sigma_{int}^2 + \frac{a^2}{4}}{b}, \tag{10}$$

$$K_{ir} = \frac{2\sigma_{int}^2 e^{-\beta}}{b} \tag{11}$$

with

$$\sigma_{int} = \sigma_d - \frac{a}{2} \tag{12}$$

$$\beta = \frac{\sigma_{int} + \frac{a}{2}}{\sigma_{int}} \tag{13}$$

The term σ_d corresponds to the desired dominant triple root for the velocity loop. Assume that the IR controller is tuned according to (9), hence, (8) is stable. On the other hand, using (12) and (13) allow the next alternative writing for Ec. (7)

$$P(s) = s^3 + as^2 + s\left(\sigma_{int}^2 + \frac{a^2}{4} - 2\sigma_{int}^2 e^{-\beta} e^{-\left(\frac{s}{\sigma_{int}}\right)}\right) + K_p\left(\sigma_{int}^2 + \frac{a^2}{4} - 2\sigma_{int}^2 e^{-\beta}\right) \tag{14}$$

Note that the quasipolynomial (14) depends only on σ_{int} and K_p. Then, it is necessary to find values of these terms to ensure the stability of the quasipolynomial (14).

A portrait of the root locus for the quasipolynomial (14) is shown in Fig.2. It is produced using the QPMR software Vyhlídal and Zítek (2014) for $K_p \in (0, 10]$ and $\sigma_{int} = 34.9015$. From this figure, it is easy to see that the quasipolynomial (14) has a double root at $s = -\sigma_{ext}$. This fact will be used in the sequel to obtain an analytical expression for K_p in terms of σ_{ext} and σ_{int}. Therefore, let $s = -\sigma_{ext}$ the desired double dominant root for the position loop. Introducing the change of variable $s \to (s - \sigma_{ext})$ into the characteristic quasipolynomial (14) yields

$$\begin{aligned}\hat{P}(s, \sigma_{int}, \sigma_{ext}, K_p) &= s^3 + s^2(a - 3\sigma_{ext}) + s(\frac{a^2}{4} + \sigma_{int}^2 \\ &-2a\sigma_{ext} + 3\sigma_{ext}^2) + K_p(\sigma_{int}^2 + \frac{a^2}{4} - 2\sigma_{int}^2 e^{-\beta}) \\ &-\sigma_{ext}\sigma_{int}^2 - \frac{a^2}{4}\sigma_{ext} + a\sigma_{ext}^2 - \sigma_{ext}^3 \\ &+2\sigma_{int}^2 e^{\left(-\frac{s}{\sigma_{int}}\right)} e^{\left(-\beta + \frac{\sigma_{ext}}{\sigma_{int}}\right)}(s + \sigma_{ext})\end{aligned} \tag{15}$$

The fact that (14) has a double root $s = \sigma_{ext}$ implies that $\hat{P}(s, \sigma_{int}, \sigma_{ext}, K_p)$ has two roots at $s = 0$. Therefore, the following holds

$$\hat{P}(s, \sigma_{int}, \sigma_{ext}, K_p) \mid_{s=0} = 0 \tag{16}$$

$$\frac{d}{ds}\hat{P}(s, \sigma_{int}, \sigma_{ext}, K_p) \mid_{s=0} = 0 \tag{17}$$

The next expression for K_p follows from (16)

$$K_p = \frac{\sigma_{ext}(\sigma_{ext}^2 - a\sigma_{ext} + \gamma - 2\sigma_{int}^2 e^{-\beta + \frac{\sigma_{ext}}{\sigma_{int}}})}{\gamma - 2\sigma_{int}^2 e^{-\beta}} \tag{18}$$

with

$$\gamma = \frac{a^2}{4} + \sigma_{int}^2 \tag{19}$$

Besides, from (17) a transcendental equation in terms of σ_{ext} follows

$$3\sigma_{ext}^2 - 2a\sigma_{ext} + \gamma - 2e^{-\beta + \frac{\sigma_{ext}}{\sigma_{int}}}(\sigma_{ext}\sigma_{int} + \sigma_{int}^2) = 0 \tag{20}$$

Note that σ_{ext} is a design parameter set beforehand. Consequently, it is necessary to solve (18) and (20) for K_p and σ_{int}. To this end, define the following

$$\sigma_{ext} = l\frac{a}{2} \tag{21}$$

$$\sigma_{int} = \left(\frac{l-1}{l}\right)k\sigma_{ext} \tag{22}$$

The term $l > 1$ is a scaling factor representing how fast is the desired position loop response with respect to the servodrive open-loop response, whereas k is a scaling factor between σ_{int} and σ_{ext}. Substituting σ_{ext} and σ_{int} given in (21) and (22) into (20) produces

$$14a^2k^2(l-1)^2 - 2e^{\frac{1}{k}-1}\left(\frac{1}{4}a^2k^2(l-1)^2 + \frac{1}{4}a^2k(l-1)l\right) + \frac{3a^2l^2}{4} - a^2l + \frac{a^2}{4} = 0 \tag{23}$$

Solving (23) for l yields

$$l = \frac{e + ek^2 - 2e^{1/k}k^2}{3e - 2e^{1/k}k + ek^2 - 2e^{1/k}k^2} \tag{24}$$

Therefore, a constant known value l, obtained from (21) by setting a value of σ_{ext}, produces a transcendental equation to be solved for k. Subsequently, using (22) it is possible to compute σ_{int}. Finally, the value of this last term allows computing K_p through (18), and K_i, K_{ir} and h using (9)-(11). The flowchart depicted in Fig.3 resumes the tuning procedure for the CPIR controller.

It is worth noting that the position and velocity loop controllers share the same value of the time delay h. On the other hand, any numerical root-finding algorithm like the Newton or bisection methods could be used for solving (23).

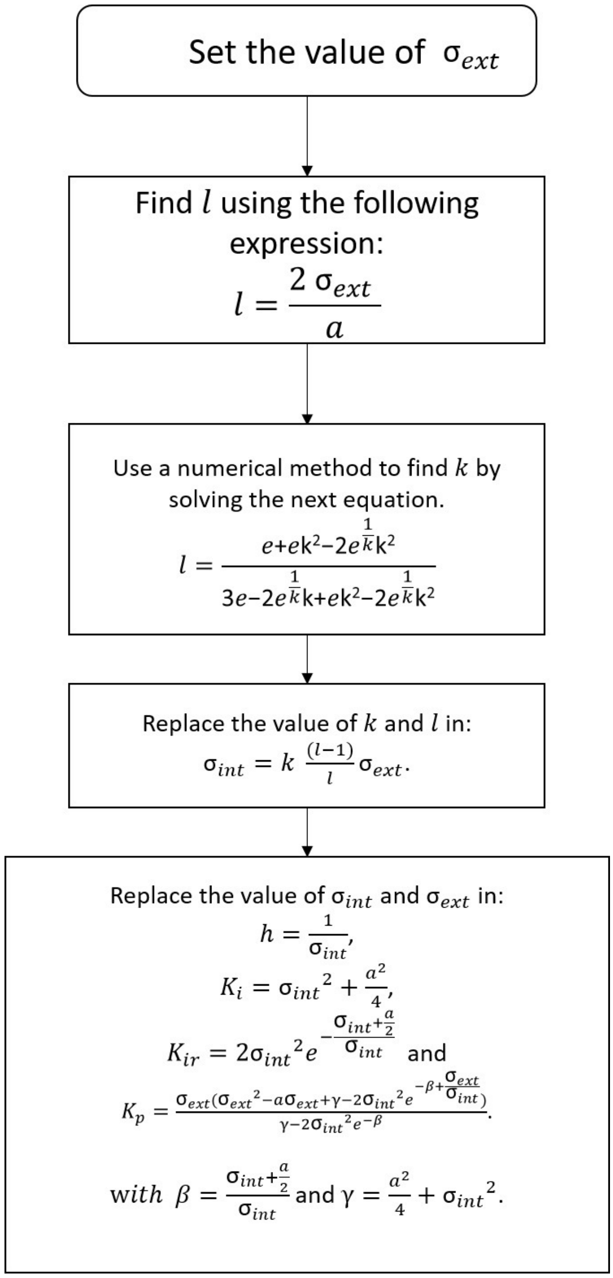

Fig. 3. CPIR tuning flowchart.

3. EXPERIMENTS

3.1 Experimental setup

Fig.(4) depicts the setup used for the experiments. The servodrive is composed of a brushed DC servomotor Clifton Precision motor JDTH-2250-BQ-IC driving a brass disk,a tachogenerator an optical encoder, and a Copley Controls power amplifier working in current mode. A ServoToGo data acquisition card STGII-8, located inside a personal computer reads the optical encoder and the voltage produced by the tachogenerator, and sends voltage control signals to the power amplifier. The Matlab/Simulink software together with the WINCON real time environment support the coding and executing of the controllers, which are implemented using the Euler method with a sampling period fixed to 1ms.

3.2 CPIR tuning

The parameters of the servodrive model (1) are $a = 0.197$ and $b = 50.98$. The CPIR controller (2) is tuned using the methodology presented in Fig.(3). In order to find the root of the transcendental equation (24), it is solved through

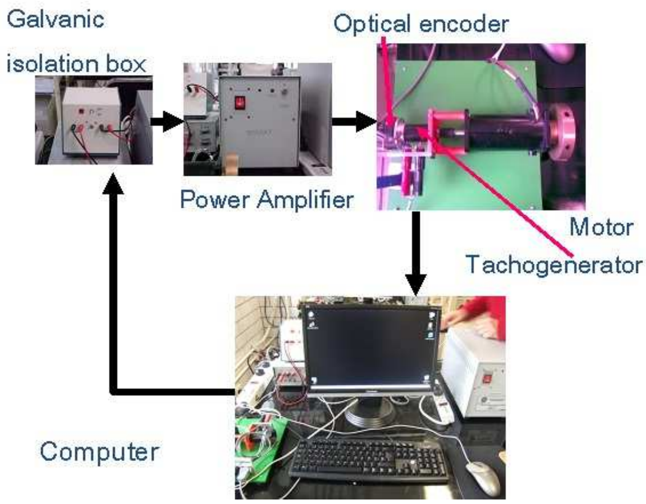

Fig. 4. Experimental setup.

Table 1. Tuning of CPIR controller applied to the servodrive.

σ_{ext}	l	k	σ_{int}
5	50.7614	3.8906	19.0697
10	101.5228	3.8623	38.2425
25	253.8071	3.8455	95.7587

Table 2. Tuning of CPIR controller applied to the servodrive.

σ_{ext}	K_p	K_{ir}	K_i	h
5	2.1389	5.2215	7.1336	0.0524
10	4.2776	21.0524	28.6872	0.0261
25	10.6937	132.2044	179.8695	0.0104

the Matlab function `fzero`, which uses a combination of bisection, secant, and inverse quadratic interpolation methods.

The results are presented in Tables (1) and (2). Note that the value of the scaling factor k is larger than 1 and remains almost constant. Note also that the proposed tuning method produces dominant roots for the velocity loop σ_{int} that are approximately four times the value of the dominant roots of the position loop defined by σ_{ext}.

3.3 Experimental results

The real-time response of the servodrive in closed loop with the CPIR controller tuned according to the proposed method is depicted in Fig.5 to Fig.7. In all the cases the step response did not show overshoots and the maximum value of the control signal and velocity response increases for large values of σ_{ext}. In order to assess the performance of the velocity loop, a variant of the CPIR controller, namely the Cascade Nonlinear Proportional Integral Retarded (CNPIR) controller, is also tested. This controller is shown in Fig 8. The main difference between the CPIR and CNPIR controllers is a saturation function applied to the position error defined by

$$\mathrm{Sat}(x) = \begin{cases} x & \text{if} \quad |x| \leq 1 \\ -1 & \text{if} \quad x < -1 \\ 1 & \text{if} \quad x > 1 \end{cases} \tag{25}$$

Hence, if the position error is small, i.e. if $|e(t)| \leq G$, then the CNPIR controller behaves like the CPIR controller.

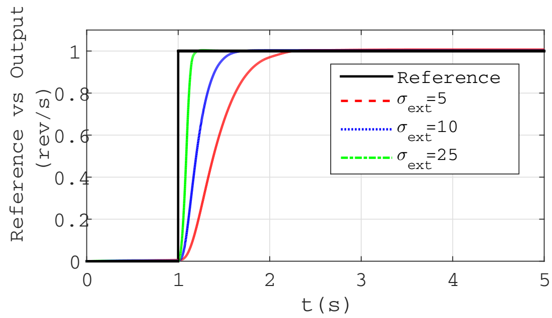

Fig. 5. Servomotor step reponses using the CPIR controller for diferent values of σ_{ext}.

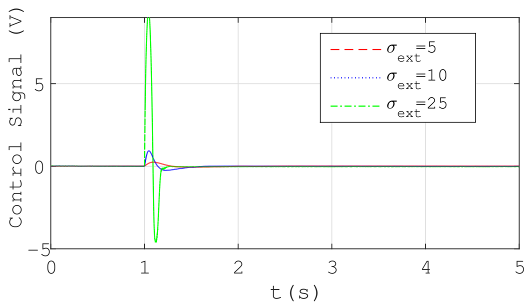

Fig. 6. Control signals for the servomotor step reponse using the CPIR controller for diferent values of σ_{ext}.

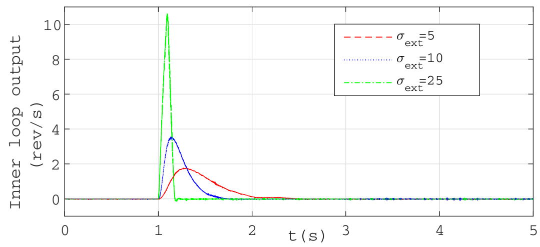

Fig. 7. Velocity response corresponding to the inner loop of the servomotor using the CPIR controller for several values of σ_{ext}.

Otherwise, the position loop is disconnected and the velocity loop works alone having a set point velocity value $1/G$.

Fig.9 displays the response of the CPIR and CNPIR controllers for a step input of 15 rev/s and $G = 0.1$. This value produces a position error large enough to enable the saturation function work in its nonlinear part, and then to make the velocity loop work alone. Note that the CPIR controller exhibits a large overshoot whereas the output generated by the CNPIR controller smoothly reach the set point value. The graphs corresponding to the control signals for both controllers is shown in Fig.10.

4. CONCLUSION

The experimental results show that the tuning procedure developed for the Cascade Proportional Integral Retarded (CPIR) controller and applied to the position control of a servodrive produces non-overshooting responses. Moreover, tuning of the CPIR controller only requires solving a nonlinear algebraic equation, a procedure easily per-

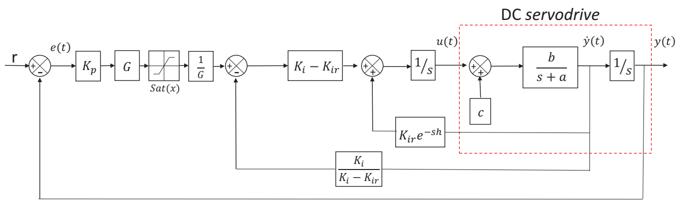

Fig. 8. Cascade Nonlinear Proportional Integral Retarded controller.

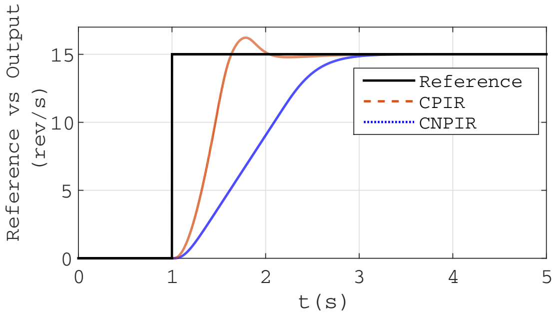

Fig. 9. Servomotor step reponses using the CPIR and CNPIR controllers for $\sigma_{ext} = 5$.

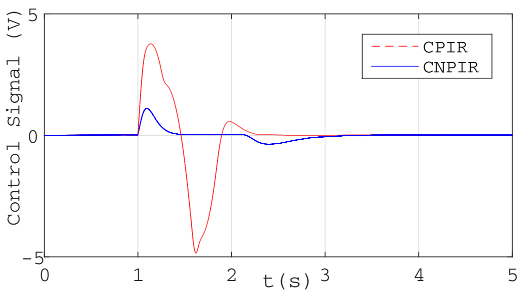

Fig. 10. Servomotor step reponses using the CPIR and CNPIR controllers for $\sigma_{ext} = 5$.

formed using standard software like Matlab or on-line solvers available in the web. Future work includes a CPIR controller analysis using an antiwindup scheme and applications to more complex systems like robot manipulators and quadrotors.

ACKNOWLEDGEMENTS

The authors would like to thank G. Castro y J. Meza for their help in setting up the laboratory prototype.

REFERENCES

Chen, Y.H. (1987). New type of controller: the proportional integral minus delay controller. *International Journal of Systems Science*, 18(11), 2033–2041.

Ellis, G. (2012). *Control system design guide: using your computer to understand and diagnose feedback controllers*. Butterworth-Heinemann.

López, K., Garrido, R., and Mondié, S. (2017). Position control of servodrives using a cascade proportional integral retarded controller. In *Control, Decision and Information Technologies (CoDIT), 2017 4th International Conference on*, 0120–0125. IEEE.

Ramírez, A., Garrido, R., and Mondié, S. (2015). Velocity control of servo systems using an integral retarded algorithm. *ISA transactions*, 58, 357–366.

Ramírez, A., Mondié, S., Garrido, R., and Sipahi, R. (2016). Design of proportional-integral-retarded (pir) controllers for second-order lti systems. *IEEE Transactions on Automatic Control*, 61(6), 1688–1693.

Suh, I. and Bien, Z. (1979). Proportional minus delay controller. *IEEE Transactions on Automatic Control*, 24(2), 370–372.

Villafuerte, R., Mondie, S., and Garrido, R. (2013). Tuning of proportional retarded controllers: theory and experiments. *Control Systems Technology, IEEE Transactions on*, 21(3), 983–990.

Vyhlídal, T. and Zítek, P. (2014). Qpmr-quasi-polynomial root-finder: Algorithm update and examples. In *Delay Systems*, 299–312. Springer.