AKADÉMIAI KIADÓ

UNIVERSITY of DEBRECEN

International Review of Applied Sciences and Engineering

DOI:
10.1556/1848.2021.00250

ORIGINAL RESEARCH PAPER

Sliding mode control of a servo system in LabVIEW: Comparing different control methods

János Máté Kiss[1], Péter Tamás Szemes[1]* and Petra Aradi[2]

[1] Department of Mechatronics, Faculty of Engineering, University of Debrecen, Debrecen, Hungary

[2] Department of Mechatronics, Optics, and Mechanical Engineering Informatics, Faculty of Mechanical Engineering, Budapest University of Technology and Economics, Budapest, Hungary

Received: January 24, 2021 • Accepted: March 26, 2021

ABSTRACT

The main contribution of this paper is to present the efficiency of LabVIEW in simulating and controlling a servo system with conventional methods (PI and PID control), as well as sliding mode control (SMC). The control of an actual system with LabVIEW and NI hardware provides an efficient implementation platform, using both LabVIEW's graphical programming and the text-based m-file language MathScript RT. Both programming environments and the connection to NI hardware are relatively easy to use, therefore, ideal for education. The graphical "coding" can help novice users to see through their algorithms. However, the mathematical background of sliding mode control is difficult compared to conventional PID control; the SMC implementation for practical uses can be quite simple, as the presented example demonstrates. The first didactic step is a simulation with the Control Design and Simulation, as well as MathScript RT Modules. Then a myRIO Student Embedded Device is used to control a real servo system. LabVIEW code can be compiled to run on computers, (soft) real-time targets, and FPGAs (hard real-time targets), so students can easily and quickly step up to real industrial measurement and control problems without the need to learn new programming environments.

KEYWORDS

DC servo motor, PI and PID control, sliding mode control, simulation, real-world system, measurement, LabVIEW, education

1. INTRODUCTION

Simulating control systems helps students understand control methods better than written textbook examples. The ability to experiment with the effects of different control strategies on real-world processes is even more helpful. Conventional PID control is part of the control engineering curriculum for bachelor-level courses. However, the more robust but mathematically far more challenging sliding mode control is usually introduced later in graduate studies.

Permanent-magnet brushed DC motors, and servo drives are widespread in industry and DIY home projects, ranging from low-power versions to vehicle drive systems. Both speed and position control are possible with a wide range of controllers, with the most common control method being PI or PID [1, 2]. PI and PID control usually satisfies the most common requirements. However, when the system has a variable load, rather than nominal ones, using a PI or PID controller does not result in a fast and stable output voltage response.

Another possible method is sliding mode control (SMC). Because of the switching behaviour of sliding mode control, it is commonly used in power electronics [3–5].

SMC is usually referred to as an interesting theory, as the mathematical model itself works very well on a theoretical level, but implementation can be a significant challenge. In the literature, some hybrid solutions are reported to overcome this problem [3, 6].

The project aims to implement PI, PID, and SMC control of a DC servo system, both as a simulation and real-world measurement. This simple system can be used as a teaching tool.

*Corresponding author.
E-mail: szemespeter@eng.unideb.hu

AKJournals

The chosen programming environment is National Instruments (NI) LabVIEW, a graphical programming environment with excellent hardware connections to real-world processes. LabVIEW programming is similar to drawing the algorithm's block diagram with user interface elements represented as nodes, programming structures such as loops, and built-in functions and procedures. The Control Design and Simulation Module consists of many VIs (Virtual Instruments, the name of LabVIEW subroutines) for control theory applications, both in time- and frequency domain. Simulation loops provide a graphical model definition, similar to the one in Simulink, and are used to the timestep by timestep simulation of various dynamical systems. The MathScript RT Module is a text-based m-file programming environment incorporated into LabVIEW. MathScript can be used as an individual application that functions similarly to MATLAB. MathScript text-based codes can also be used within a graphical LabVIEW program when inserted into a so-called MathScript Node that acts as a VI.

LabVIEW can interface with measurement and automation products from various manufacturers, besides devices from LabVIEW's developer National Instruments. NI has specific hardware products for students, such as the myRIO Student Embedded Device that was chosen for the measurement and control of the real-world DC servo system. The servo system was first simulated in LabVIEW. PI, PID, and SMC control methods were programmed in LabVIEW/MathScript, and various experiments were performed to evaluate the control strategies. Implementing the simulation and measurement setup is relatively simple; the programming environment is clear-cut and self-explaining. LabVIEW does not require long to learn; therefore, it is excellent for demonstrational purposes in higher-level education. There are even versions of LEGO robots, e.g. Robotics Invention System that shipped with a special LabVIEW version. Further gain by using LabVIEW is that the developed controller code can be compiled to run not just on computers but (soft) real-time targets and FPGAs (hard real-time targets). That way, students can easily and quickly step up from computer simulation to real industrial measurement and control problems without the need to learn new programming environments.

The presented example of the sliding mode control is far from the complex industrial applications; however, the measurement setup provides an excellent possibility to introduce the basics of this control method with an existing physical system even for bachelor-level students. The DC-motor system can further be used to test other control strategies both with the simulated and the actual process.

2. PI AND PID CONTROLLERS

The PI or PID controller is the most used solution for controlling the speed of motors. Their operation is based on error signal compensation. They consist of a total of two or three parallelly connected terms whose initials give their name. P stands for proportional, I for integrating, and D for derivative. The output of the proportional term is proportional to the error signal, the integrating term is proportional to the integral of the error signal, and the derivative term is proportional to the derivative of the error signal. As the gain of the proportional member increases, the value of the control signal and with it the error increase proportionally. The controller will try to respond faster to the error signal, but at the same time, the overshoot will also increase. With the help of the integrating member, not only the rise time but also the steady-state error can be reduced. The disadvantage is that it can slow down the system and cause oscillations when the sign of the error signal changes because it may take some time to follow it. The output of the PI controller is described by Eq. (1).

$$u(t) = K_p \cdot e(t) + K_i \cdot \int_0^t e(\tau)\mathrm{d}\tau \tag{1}$$

K_p is the gain of the proportional term, e is the error signal given by the difference between the setpoint (reference signal) and the process variable, K_i is the gain of the integrating term, and τ is the integration time.

By adding a derivative term, the controller can predict the error in advance. It can be used to amplify the control signal while keeping the amplitude of the error relatively small. This will dampen the system and reduce overshoot while not affecting the steady-state error. The output of a complete PID controller supplemented with a derivative term is described by Eq. (2), where K_d is the gain of the derivative term.

$$u(t) = K_p \cdot e(t) + K_i \cdot \int_0^t e(\tau)\mathrm{d}\tau + K_d \cdot \frac{\mathrm{d}e(t)}{\mathrm{d}t} \tag{2}$$

There are other mathematical representations of the PID-controller's transfer function; in LabVIEW the transfer function version of Eq. (2) is called Parallel, with the following parameters:

$$K_c + \frac{1}{T_i s} + T_d s \tag{3}$$

The other two formats are Academic:

$$K_c\left(1 + \frac{1}{T_i s} + T_d s\right) \tag{4}$$

and Series:

$$K_c\left(1 + \frac{1}{T_i s}\right)(1 + T_d s) \tag{5}$$

The low-pass filter of the derivative term can be specified with the α parameter, in each representation, when appropriate:

$$\frac{1}{\alpha T_d s + 1} \tag{6}$$

3. SLIDING MODE CONTROL

Sliding mode control was developed in the Soviet Union primarily for aerospace and missile applications in the 1970s [7, 8]. The mathematical basis of SMC design can be found in [9].

Although understanding the exact mathematical description behind the method can be challenging, in many cases, it is relatively easy to apply SMC without a deeper knowledge of its complex mathematical background. Because of this and its robustness, it is widespread, for example, in robotics, power electronics, and servo drive control, where variable system structures are common [3–5, 10].

SMC's aim is to bring the system into a state in which its dynamics can be described by a differential equation with a lower degree of freedom, in which case, in theory, the system is completely independent of changes in certain types of parameters and certain types of external disturbances. This condition is called sliding mode. Although, according to the theory, sliding mode control appears to be a well-functioning and robust control method, unfortunately, there are severe limitations to its practical implementation. The main problem is the so-called chattering [11], which means a high-frequency oscillation around the sliding surface that significantly reduces the efficiency and robustness of the controller. While, in theory, SMC provides superior performance of the closed-loop system in sliding mode, the practical limitations discourage some researchers. The need for higher sampling frequency than other control strategies to reduce the high-frequency oscillation phenomenon (chattering) is the limiting factor. There are approaches to overcome this problem, such as an observer-based, discrete-time, sliding mode control design that prevents the system from entering the critical domain [12, 13].

The first step in designing a sliding mode controller is to define the sliding surface. The sliding surface is as follows:

$$s = \left(\frac{\mathrm{d}e(t)}{\mathrm{d}t} + C \cdot e(t)\right)^{n-1} \tag{7}$$

s is the sliding surface, C is a strictly positive constant that determines the bandwidth of the system (also mentioned as λ in some literature), n is the degree of the system, and e is the error signal. The transfer function of the brushed DC motor is second degree, so by substituting the speed, the sliding surface is obtained as follows:

$$s = \frac{\mathrm{d}\omega_e(t)}{\mathrm{d}t} + C \cdot \omega_e \tag{8}$$

Rotational speed ω_e is the error signal defined as the difference between the reference signal and the process variable. Once the sliding surface has been determined, the next step is to create a control signal with which the sliding surface can be reached and maintained. This is subject to:

$$s \cdot \dot{s} > 0 \tag{9}$$

In order to satisfy this condition, the principle of the discontinuous controller output signal is obtained with the sign function, where the variable is the instantaneous value of the sliding surface and K is a positive constant:

$$u = K \cdot sgn(s) \tag{10}$$

The sign (sgn) function in (6) is defined as

$$s = \begin{cases} -1, & s<0 \\ +1, & s \geq 0 \end{cases} \tag{11}$$

The use of the sign function may lead to chattering, which can have a harmful effect on the motor, so avoiding this phenomenon is very important for SMC. One method to prevent chattering is to replace the sign function with the pseudo function:

$$u = K \cdot \frac{s}{|s| + \delta} \tag{12}$$

where δ is a small positive constant called a tuning parameter to reduce the chattering [14]. The choice of δ is a crucial consideration because if it is too small, the chattering may still be present, but if it is too large, reaching the reference value may cause a problem to the controller [15].

Figure 1 shows the MathScript implementation of the sliding mode controller as a LabVIEW MathScript Node with inputs on the left side and the controller output on the right side of the frame.

4. LABVIEW MOTOR MODEL

The transfer functions of the DC motor's electrical and mechanical characteristics can be created using the built-in function block of the Construct Transfer Function Model VI. The numerator of the transfer function is given by the constant value of 1, and the denominator consists of an indexed array, the elements of which are read from the cluster containing the motor parameters in SI units (Table 1). The symbols are as follows: armature resistance and inductance are denoted by R_a and L_a, respectively. J is the moment of inertia, and B represents the friction coefficient. K_v is the back-EMF constant, and K_t is the torque constant. The transfer function elements are listed in descending order of degree. In the next step, the generated transfer function is converted by the Convert Control Design to Simulation VI function block to a form that can be used in the simulation loop (Fig. 2).

The block diagram of Fig. 2 represents the transfer function between the motor's voltage input and rotational speed output (the integrator labelled Position calculates the

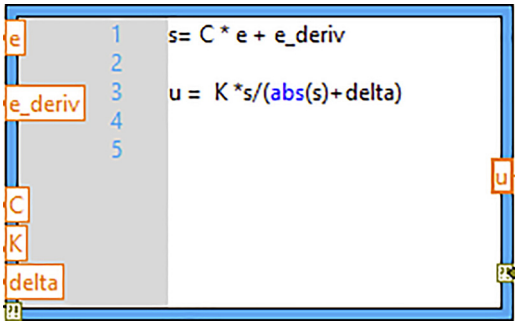

Fig. 1. Calculation of the control signal of the sliding mode controller

Table 1. Parameters of motors for the transfer function

Parameter	Test1	Test2	Test3
Ra	2	0.6	1
La	0.5	0.8	0.2
Kt	0.1	0.2	0.2
Kv	0.1	0.2	0.8
J	0.2	0.9	0.4
B	0.02	0.7	0.5

integral of rotational speed and is not included in the following transfer function):

$$G(s) = \frac{K_t}{(Ls + R)(Js + B) + K_t K_v} \tag{13}$$

Each parameter can be saved or loaded in XML format for the repeatability of the simulations. The XML file's path can be input via the program's user interface or hard-coded into the program. The parameters can be validated by pressing the apply button.

The model's correct behaviour can be tested with several test functions available in LabVIEW, or the user can create an own test VI. A function generator has been placed in the program, which can be freely configured in the user interface to create several general test signals. Configurable parameters are duty cycle, period, amplitude, and offset. An Express VI is also included in the program to save the simulation data to an xlsx spreadsheet file. The current time of the simulation, the current value of the test signal, the motor speed, the current, and the position is stored in the file after each iteration.

The simulation results obtained with each parameter are shown in Fig. 3.

5. SIMULATION RESULTS

The controller parameters were determined with LabVIEW's Control Design and Simulation Module's built-in tuning blocks for the PI and PID controllers with the use of various available tuning methods (ZN: Ziegler-Nichols, CC: Cohen-Coon, CHR: Chien-Hrones-Reswick, IMC: Internal Model Control). The details of these tuning methods are available in the official user manual and help.

The controller parameters determined by Internal Model Control are used, as this method gave the fastest operation, according to the simulation results. The parameters of the PI and PID controllers are shown in Table 2.

The parameter values of the sliding mode controller are C = 10, K = 40 and δ = 0.05.

5.1. Step function

The first test for each controller was a step function test in which the speed setpoint jumped from 0 to 10 rad/s in the first second of the simulation. The speed responses obtained for this are summarized in Fig. 4.

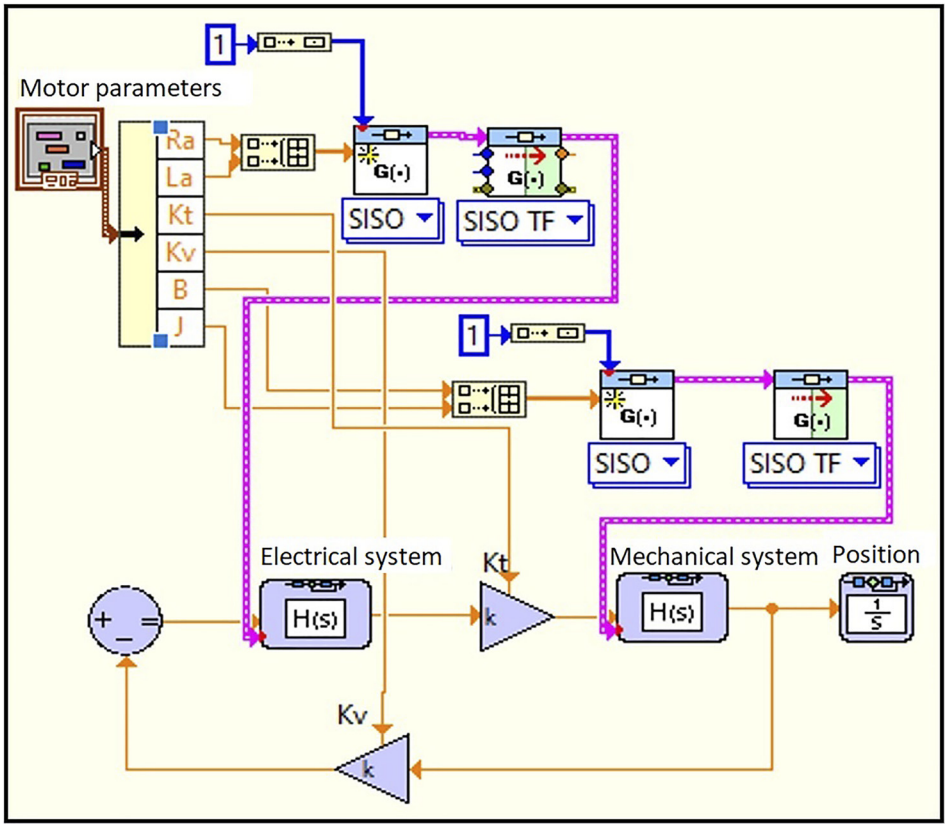

Fig. 2. Block diagram of the model

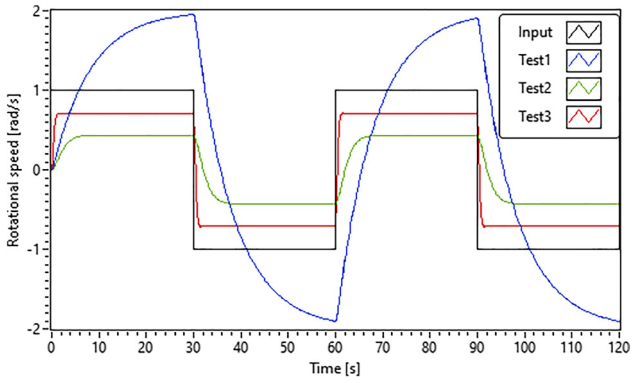

Fig. 3. Results of motor model testing with different parameters

The simulation results in Table 3 show that the theoretical differences between controllers are significant. As expected, the PI controller is the slowest and has the most extensive overshoot. The PID, which has an additional derivative member, produced a slightly smaller overshoot, and both the rise and settling times are shorter. Of the three controllers, the sliding mode controller performed outstandingly well. Both the rise and settling times are quicker than the PI and PID controllers, and there is no overshoot.

The phase margin value was nearly identical for the PI and PID controller: 53.36° for PI and 53.73° for PID.

Figure 5 shows the control signal for different controllers with the characteristic alternating in the sliding mode controller's control signal and the PID controller's relatively large initial value.

Figure 6 shows the error and error derivative plots of the different controllers with the two distinctive parts for the sliding mode controller.

5.2. Tracking

In this case, the initial value of the speed reference signal is 10 rad/s, and after the steady-state is reached, it is changed abruptly to 5 rad/s in the fifth second. The comparison is shown in Fig. 7. The results again reflect the difference described in theory as in the previous case of the step function. The sliding mode control was spectacularly better in this study as well, as it was able to follow the change of the reference signal the fastest and to reach the new steady-state asymptotically. The tracking results show similar performance to step response results. Sliding mode control performed best; the undershoot and settling time of the loop with the PID controller were smaller than those of the one with the PI controller.

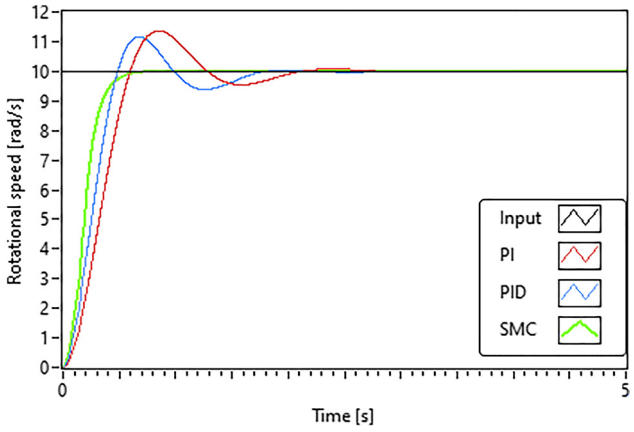

Fig. 4. Comparison of controllers with step function

Table 3. Simulation results

Controller	Rise time (s)	Settling time (s)	Overshoot (%)	Undershoot (%)
PI	0.395	1.977	13.56	4.97
PID	0.326	1.687	11.61	5.17
SMC	0.281	0.591	–	–

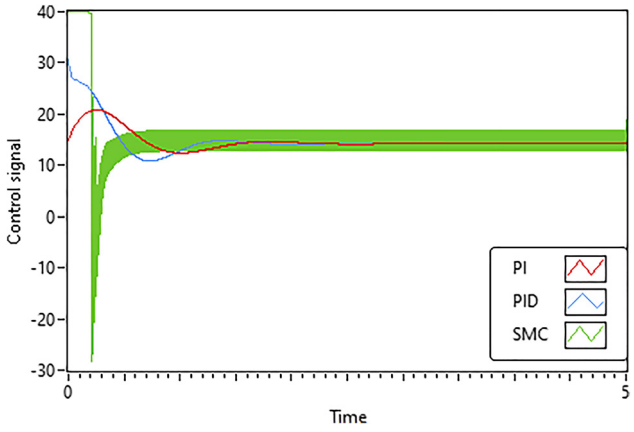

Fig. 5. The control signal of different controllers

5.3. Speed function

In the case of the speed function, the result is similar to the previous ones; the sliding mode controller performed best, as it was able to follow the signal almost perfectly in the case of the ideal motor model. The difference between PI and PID controllers is as expected from the theory. The response

Table 2. Parameters of controllers

Parameter	Controller	ZN	CC	CHR	IMC
Kc	PI	2.120	2.228	1.413	2.356
	PID	2.635	3.466	2.238	6.011
Ti (min)	PI	0.012	0.006	0.014	0.006
	PID	0.007	0.007	0.008	0.008
Td (min)	PI	–	–	–	–
	PID	0.002	0.001	0.001	0.001

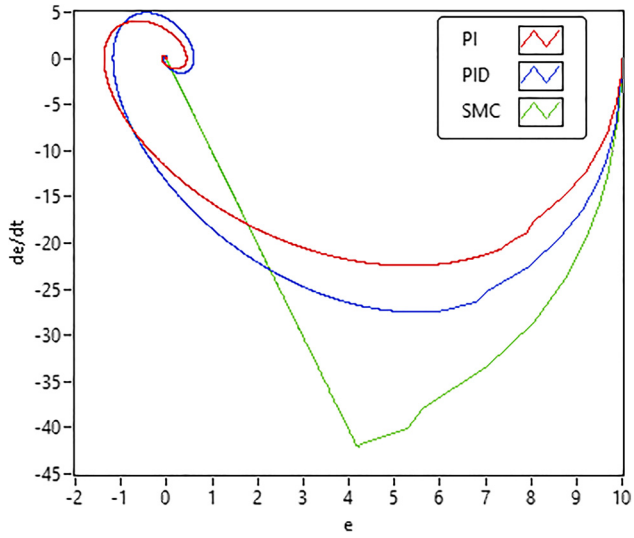

Fig. 6. Error and derivative of the error

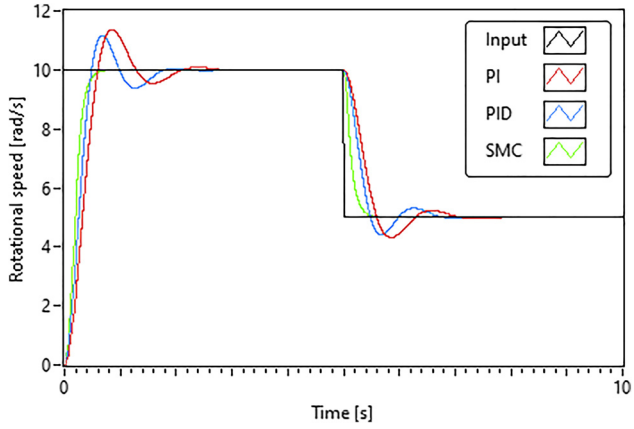

Fig. 7. Comparison of controller's signal tracking capability

of the controllers is shown in Fig. 8. For the comparison, the results of the best performers were used, so in the case of the PI controller, the curves obtained with the smallest average error are Cohen-Coon, while in the case of the PID controller, the curves were obtained with the parameters of the Internal Model Control method.

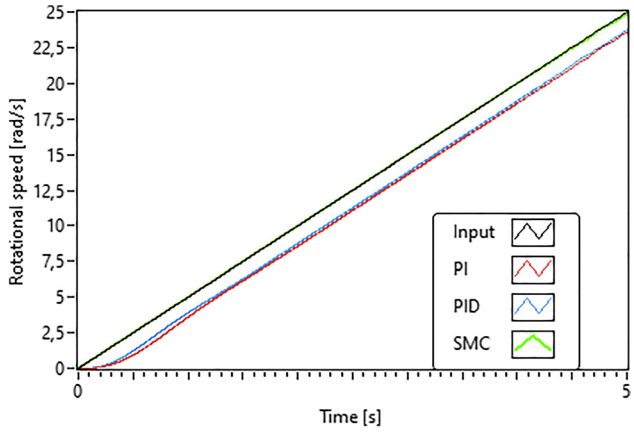

Fig. 8. Speed response of the controllers

5.4. Sine test signal

Similar conclusions can be drawn from the curves for the sinusoidal test signal as for the speed function. Here, however, for both PI and PID controllers, the parameters of Internal Modern Control proved to be the most efficient in tracking the sinusoidal signal. In Fig. 9, the angular velocity or rotational speed response curve obtained with the sliding mode controller almost wholly covers the reference signal, the difference being noticeable only in a few places.

5.5. Summary

Overall, similar results were obtained for all four test scenarios, according to which the sliding mode control produced much better results at the brushed DC motor speed control. The theoretical differences between the three different controllers were also well visible in the simulations. By choosing the appropriate parameters, it was possible to make the sliding mode controller work more efficiently than the PI and PID controllers while also eliminating the chattering phenomenon, which did not happen in any of the tests. The summary of control performance indicators are shown in Table 4 and 5. Smaller number means better control performance. According to the results, the SMO has the best control performance against PI and PID control laws.

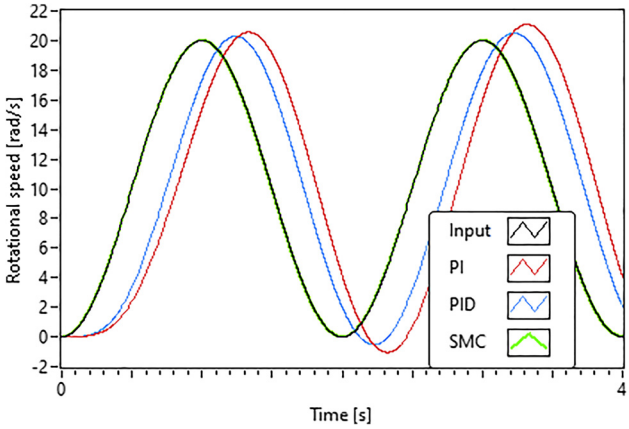

Fig. 9. Sine test signal response comparison

Table 4. Error results of controllers

Controller	Average error (rad/s)	Average error (%)	Largest error (rad/s)	Largest error (%)
PI	0.91	6.35	0.91	6.35
PID	0.48	3.15	0.48	3.15
SMC	0.02	0.12	0.02	0.12

Table 5. Error results of controllers

Controller	Average error (rad/s)	Largest error (rad/s)
PI	5.65	9.42
PID	3.79	6.35
SMC	0.07	0.43

6. MEASUREMENT AND CONTROL SYSTEM WITH A REAL MOTOR

The main components of the physical system are the power supply, the NI myRIO Student Embedded Device, that serves as the interface between the real-world system and the computer running the LabVIEW code, the motor drive circuit, the permanent magnetic brush DC electric motor, and the speed feedback encoder. The assembly is illustrated in Fig. 10. The myRIO has dedicated inputs corresponding to terminals A and B of the quadratic encoder, which are digital inputs DIO18 and DIO22. It also contains dedicated outputs for the PWM signal, for which the digital output DIO27/PWM0 has been used. A 5V high signal is required on the motor drive circuit to turn on the motor, which is specified at output DIO13.

The main parameters of the motor are summarized in Table 6.

Figure 11 shows part of the LabVIEW program used in the experimental setup. The encoder's signal represents the process value. The reference signal and process value are the inputs of the PID controller block, together with the PID parameters, that come from the tuning process. The PID block's output is the control signal connected to the DC-motor's input via the PWM block. This typical LabVIEW "program code" illustrates the ease of use and transparency of LabVIEW applications with connected hardware components via a connecting device, in this example, a myRIO Student Embedded Device.

7. RESULTS WITH THE REAL MOTOR

Real-world measurements differ from the simulation in motor parameters that result in different timescales, test signals, and instead of rotational speed in rad/s, RPM in 1/min is used. It is somewhat intentional because students are required to analyse results and implement the simulation with the experiments' parameters, not just check whether the simulated results look like the real-world measurements.

Table 6. Parameters of the real motor

Resistance [Ω]	2.32
Inductance [mH]	0.238
Speed constant [(1/min)/V]	408
Torque constant [m Nm/A]	23.4
Rotor inertia [g cm^2]	10.8

In the real DC-motor system PI and PID controllers are implemented with the built-in VIs, like in the case of the ideal motor model. It is important to note that the step size of the simulation and the step size of the PID VI must be consistent so that the instantaneous speed is calculated correctly from the encoder signal. The definition of the parameters and the operation of the program are the same as those used for the ideal model. The parameters determined for each controller with the use of various tuning methods are summarized in Table 7.

The sliding mode control parameters this time are C = 2, K = 1 and δ = 0.9. The value of K determines the range of the output and since the duty cycle can take a value between 0 and 1 according to the Express VI used, it also determines the value of the parameter. The value of C was determined empirically, in the case of further increase, the control was not functional, and the value of δ still had to be chosen to be 0.9.

Responses to various test scenarios are shown in Figs 12–15. It is important to note that the results seem to show the effect of saturation that was not present in the simulation.

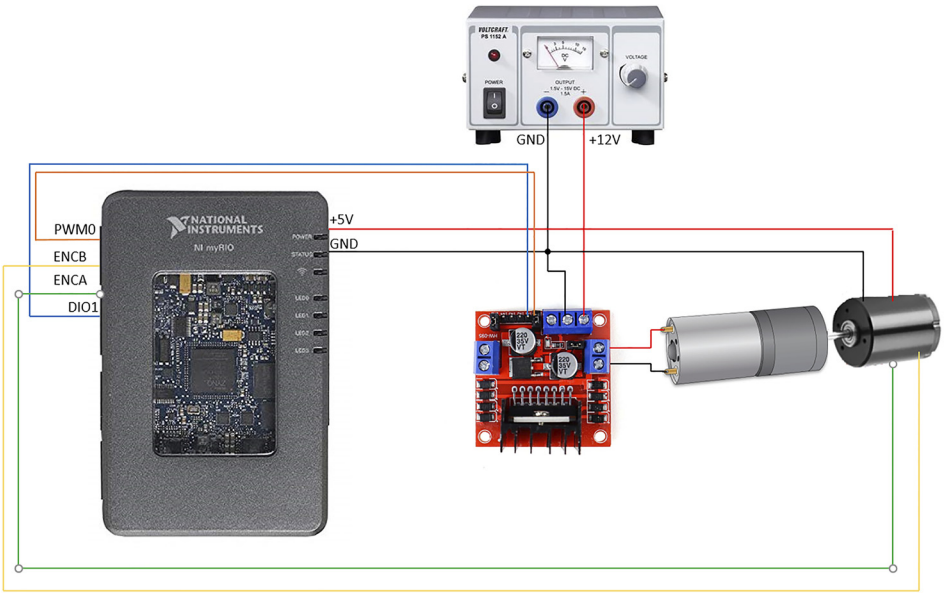

Fig. 10. The measuring system

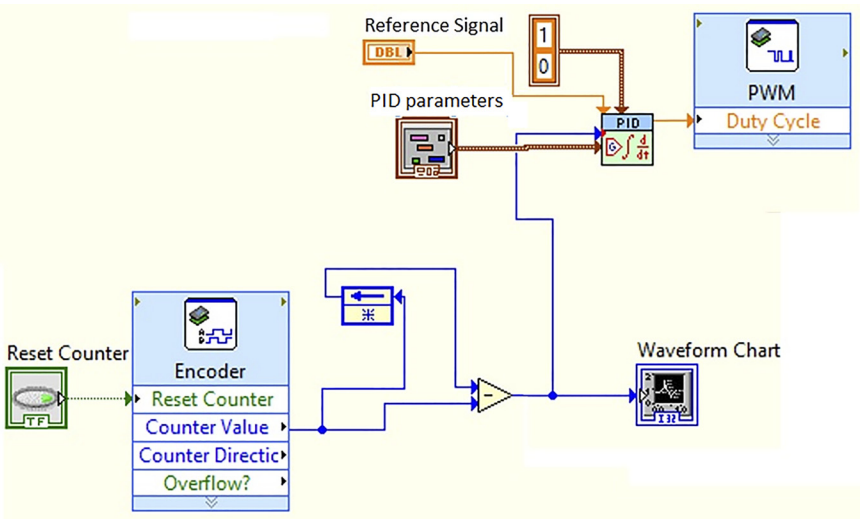

Fig. 11. Part of the real-world DC-motor control LabVIEW Block Diagram

Table 7. Parameters of controllers

Parameter	Controller	ZN	CC	CHR	IMC
Kc	PI	0.39	0.40	0.26	0.44
	PID	0.52	0.59	0.41	0.90
Ti (min)	PI	1.34E-05	1.18E-05	1.61E-05	6.09E-05
	PID	8.04E-06	9.62E-06	9.65E-06	6.29E-05
Td (min)	PI	–	–	–	–
	PID	2.01E-06	1.44E-06	1.69E-06	1.95E-06

7.1. Step function

The responses to the step function measured by the controllers are as expected. The Ziegler-Nichols parameters gave the best results for the PI controller and the Cohen-Coon parameters for the PID controller. According to the theory, the PID controller gave a more favourable result in terms of overshoot. There is no significant difference in rise-time and settling time. The sliding mode controller was able to achieve better results without overshoot, with the steady-state settling in nearly a third of that time. The results obtained are shown in Table 8.

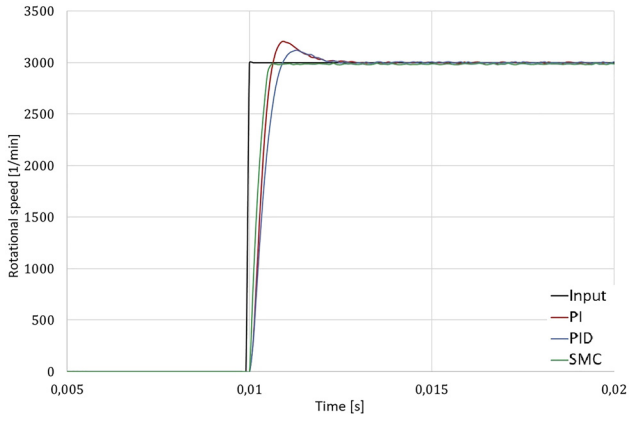

Fig. 12. Comparison of controllers based on step function with a real motor

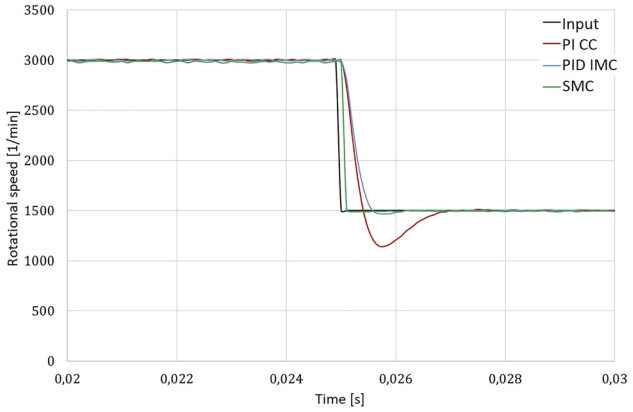

Fig. 13. Comparison of controllers in terms of signal tracking with a real motor.

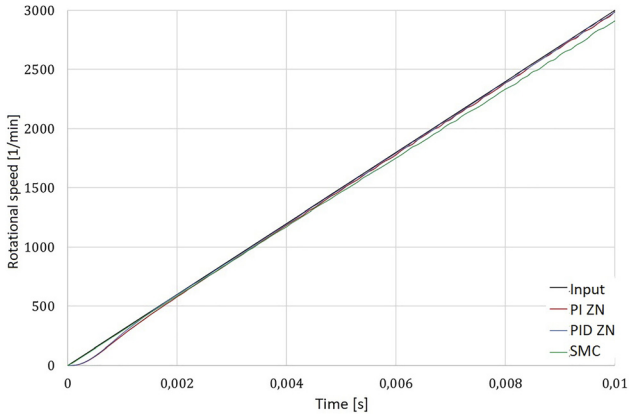

Fig. 14. Comparison of controllers based on speed function with a real motor

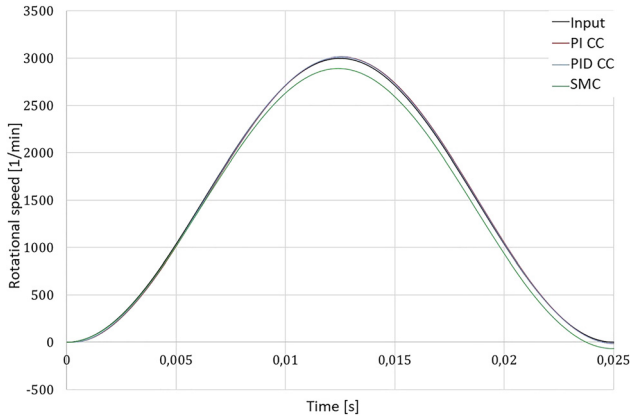

Fig. 15. Comparison of controllers with sinusoidal test signal with real motor

7.2. Tracking

In tracking the abrupt change of the reference signal. the PID controller performed surprisingly well with the parameters of the Internal Model Control method compared to the other results.

The sliding mode controller performed best, followed by the PID and then the PI in the last place. Table 9 shows the characteristics of the responses.

7.3. Speed function

For the speed function test signal. the Ziegler-Nichols parameters gave the best results for the PI and PID controllers.

After the start. the error is roughly constant with the PI and PID controllers. while SMC's error increases more and more with the increase of the input signal. The properties of the response signals are shown in Table 10.

7.4. Sine signal

In the case of the sinusoidal test signal, a better measurement result was obtained with the Cohen-Coon parameters at least, however, this difference may also be due to the measurement error mentioned earlier.

Table 8. Results of step input tests

Controller	Rise time (s)	Settling time(s)	Overshoot (%)	Undershoot (%)
PI	0.0005	0.0017	7.17	–
PID	0.0018	4.01	–	–
SMC	0.0004	0.0006	–	–

Table 9. Results of tracking tests

Controller	Settling time (s)	Undershoot (%)	Overshoot (%)
PI	0.0017	24.30	0.78
PID	0.0009	4.40	–
SMC	0.0001	–	–

Table 10. Results of speed function tests

Controller	Average error (rad/s)	Average error (%)	Largest error (rad/s)	Largest error (%)
PI	16.50	1.12	71.39	47.56
PID	12.37	0.83	63.38	52.72
SMC	40.00	2.67	88.80	3.03

Barely noticeable differences can be seen in Fig. 15; details are presented in Table 11.

8. SUMMARY

When comparing the results of the performed experiments, the sliding mode control worked excellently in the case of the ideal theoretical model. For all experiments, SMC was able to follow the reference signal extremely well for both abrupt changes and continuously changing signals. In the case of PI and PID controllers, there was a significant difference in the parameters determined by different controller tuning methods, although not all test signals gave the best results.

With the real motor, the sliding mode controller no longer had a clear advantage. Although in cases where there was an abrupt change in the reference signal value it has performed much better, in the case of continuously changing signals such as the speed function or the sine signal it has already lagged behind the other two controllers.

Simulation is a very useful tool as students can check various controller parameter settings, comparing the results, and, therefore, understand the role of the PID-controller's parts and their interaction. Experimenting with the tuning of a simulated control system often provides deeper knowledge than the use of controller tuning methods from textbooks without understanding the underlying principles.

The problems of saturation and integral windup usually occur in real-world physical systems. There is an educational value of showing these problems with relatively small-scale real-world systems without the risk of serious damage, as well as including the appropriate simulated versions beyond the idealistic simulated system without considering saturation and integral windup. The detailed discussion of saturation and integral windup is going to be included in the laboratory experiments' handouts soon.

The whole process of modelling the DC servo system, designing controllers and simulating the closed-loop control system, and later performing experiments in a real-world system based on the simulation results has a significant educational value. Control engineering is considered a rather

Table 11. Results of sine signal tests

Controller	Average error (rad/s)	Largest error (rad/s)
PI	16.60	26.66
PID	11.42	18.47
Sliding mode	89.81	149.15

difficult subject all over the world [16, 17]; therefore, it is well worth introducing the applications of the theory with experiments like the one presented above. As the pandemic situation enforces distant education worldwide, the value and importance of simulation increases. Students are not allowed to participate in laboratory experiments, so the chance of using real-world devices is reduced. Educational devices, such as the presented NI myRIO, offer the possibility of self-paced experiments. especially when students can gain access to them by borrowing or use via the Internet. The expansion of the presented system to Internet access is currently being investigated.

ACKNOWLEDGEMENT

This research was funded by TKS2020-NKA-04. Project no. TKP2020-NKA-04 has been implemented with the support provided from the National Research, Development and Innovation Fund of Hungary, financed under the 2020-4.1.1-TKP2020 funding scheme.

REFERENCES

[1] T. Wildi, *Electrical Machines. Drives and Power Systems*. Upper Saddle River. NJ. USA: Pearson Prentice Hall, 2006.

[2] G. Sziebig, B. Takarics, and P. Korondi, "Control of an embedded system via Internet," *IEEE Trans. Ind. Electron.*, vol. 57, no. 10, pp. 3324–33, Oct. 2010, https://doi.org/10.1109/TIE.2010.2041132.

[3] K. Al-Hosani, A. Malinin, and V. I. Utkin, "Sliding mode PID control of buck converters," in *2009 European Control Conference (ECC)*, Budapest, 2009, pp. 2740–4, https://doi.org/10.23919/ECC.2009.7074821.

[4] P. Korondi and H. Hashimoto, "Park vector based sliding mode control of UPS with unbalanced and nonlinear load," in *Variable Structure Systems. Sliding Mode and Nonlinear Control*, K D. Young and Ü. Özgüner, Eds., London. UK: Springer, 1999, pp. 193–209.

[5] P. Korondi, S. H. Yang, H. Hashimoto, and F. Harashima, "Sliding mode controller for Parallel resonant dual converters," *J. Circuits. Syst. Comput.*, vol. 5, no. 4, pp. 735–46, 1995. https://doi.org/10.1142/S0218126695000424.

[6] R. E. Precup, S. Preitl, and P. Korondi, "Fuzzy controllers with maximum sensitivity for servosystems," *IEEE Trans. Ind. Electron.*, vol. 54, no. 3, pp. 1298–310, 2007. https://www.doi.org/10.1109/TIE.2007.893053.

[7] V. I. Utkin and K. D. Yang, "Methods for construction of discontinuity planes in multidimensional variable structure systems," *Autom. Remote Control*, vol. 39, no. 10, pp. 1466–70, 1979.

[8] K. D. Young, "Controller design for a manipulator using theory of variable structure systems," *IEEE Trans. Syst. Man. Cybernetics*, vol. 8, no. 2, pp. 101–9, Feb. 1978, https://doi.org/10.1109/TSMC.1978.4309907.

[9] K. Széll and P. Korondi, "Mathematical basis of sliding mode control of an uninterruptible power supply," *Acta Polytech. Hungarica.*, vol. 11, no. 3. 2014. https://doi.org/10.12700/aph.11.03.2014.03.6.

[10] P. Korondi and H. Hashimoto, "Sliding mode design for motion control," in *Applied Electromagnetics and Computational Technology II : Proceedings of the 5th Japan-Hungary Joint Seminar on Applied Electromagnetics in Materials and Computational Technology*. IOS Press, 2000, pp. 221–32.

[11] V. Utkin and H. Lee, "Chattering problem in sliding mode control systems," *IFAC Proc. Volumes*, vol. 39, no. 5, p. 1, 2006. https://doi.org/10.3182/20060607-3-IT-3902.00003.

[12] P. Korondi, H. Hashimoto, and V. Utkin, "Discrete sliding mode control of two mass system," in *1995 Proceedings of the IEEE International Symposium on Industrial Electronics*, Athens, Greece, vol. 1, 1995, pp. 338–43, https://doi.org/10.1109/ISIE.1995.497019.

[13] P. Korondi, H. Hashimoto, and V. Utkin, "Direct torsion control of flexible shaft in an observer-based discrete-time sliding mode," *IEEE Trans. Ind. Electron.*, vol. 45, no. 2, pp. 291–6, April 1998, https://doi.org/10.1109/41.681228.

[14] H. Maghfiroh, A. Sujono, M. Ahmad, A. Brillianto, and H. Chico, "Basic tutorial on sliding mode control in speed control of DC-motor," *J. Electr. Electron. Inf. Commun. Technol.*, vol. 2, no. 1, pp. 1–4, 2020. https://www.doi.org/10.20961/jeeict.2.1.41354.

[15] E. H. Dursu and A. Durdu, "Speed control of a DC motor with variable load using sliding mode control," *Int. J. Comput. Electr. Eng.*, vol. 8, no. 3, pp. 219–26, 2016. https://www.doi.org/10.17706/IJCEE.2016.8.3.219-226.

[16] P. Albertos and I. Mareels, *Feedback and Control for Everyone*. Berlin, Heidelberg, Germany: Springer, 2010.

[17] K. Zenger, "Control engineering. system theory and mathematics: the teacher's challenge," *Eur. J. Eng. Edu.*, vol. 32, no. 6, pp. 687–94, 2007. https://doi.org/10.1080/03043790701520719.