Software Impacts 13 (2022) 100306

Contents lists available at ScienceDirect

Software Impacts

journal homepage: www.journals.elsevier.com/software-impacts

Original software publication

Digital implementation of the PID controller

Pujan Bhandari [a,*], Péter Zoltán Csurcsia [a,b]

[a] Vrije Universiteit Brussel, Department of Engineering Technology, Belgium
[b] Brussels Institute for Thermal-fluid systems and clean Energy (BRITE), Pleinlaan 2, B-1050 Elsene, Belgium

ARTICLE INFO

Keywords:
PID controllers
Digital implementation
Closed-loop control
Industrial applications

ABSTRACT

This paper discusses the appropriate way of implementing the PID controller in the software. This work can be seen as a tutorial that teaches the important concepts of the PID controller. The PID controller equation presented in most undergraduate textbooks is in the continuous Laplace domain. To implement the PID controller in the software the continuous Laplace domain equation must be transformed into the difference equation. Once transformed into a difference equation, this equation can be implemented to develop a digital PID controller to control any closed-loop system. This concept is discussed in this paper with the help of software that is developed to control a ball balancing beam.

Code metadata

Current code version	v1.2
Permanent link to code/repository used for this code version	https://github.com/SoftwareImpacts/SIMPAC-2022-29
Permanent link to reproducible capsule	
Legal code license	GNU General Public License (GPL)
Code versioning system used	none
Software code languages, tools and services used	python
Compilation requirements, operating environments and dependencies	python 3.8 or higher
If available, link to developer documentation/manual	http://sysidguy.eu/?page=pingpong&l=4
Support email for questions	peter.zoltan.csurcsia@vub.ac.be, pujan.bhandari@vub.be

1. Introduction

This paper presents the process of designing the discrete-time PID controller from scratch. There are various libraries available that employ the PID controller [1]. However, in the cases where the libraries are not available, the PID controller must be designed manually with the basic knowledge of system-control theory. In general, the systems are described in the continuous domain and assumed to be linear [2] and time-invariant [3]. The hydraulics systems, pneumatic systems, or analog electronics were used to realize the controllers, but nowadays, these are almost fully substituted by digital processors [4]. Thus, the signals should be quantified to be able to use the digital processors to design the controller. The continuous-time Laplace equation describing the PID controller is $\frac{C(s)}{E(s)} = K_C \cdot [1 + \frac{1}{\tau_I \cdot s} + \tau_D \cdot s]$. This equation cannot be implemented directly to the discrete-time digital processor, but it must be approximated by a difference equation [5]. This can be done mainly in two steps: the transformation of the Laplace domain equation into the z-domain and the transformation of the z-domain equation into the difference equation.

This work is developed to demonstrate the application of the PID controller in real life and discuss the possible challenges encountered during its digital implementation. The demonstration setup that this software control is a ball balancing beam. The core goal of this software is to make it compatible with a cost-effective demonstration setup that can be used for teaching purposes for undergraduate students. The physical setup consists of a camera as a sensor, an Arduino as a microprocessor, a servomotor as an actuator, and the beam and a ball itself as the system.

2. Development of the software

The software has been entirely developed in Python programming language as it is well known among engineering students. Further, it also does not require any license which makes it accessible to anyone in every part of the world. The following subsections discuss how the software is [1] exactly developed.

[*] Corresponding author.
E-mail addresses: pujan.bhandari@vub.be (P. Bhandari), peter.zoltan.csurcsia@vub.ac.be (P.Z. Csurcsia).

https://doi.org/10.1016/j.simpa.2022.100306
Received 16 March 2022; Received in revised form 28 April 2022; Accepted 28 April 2022

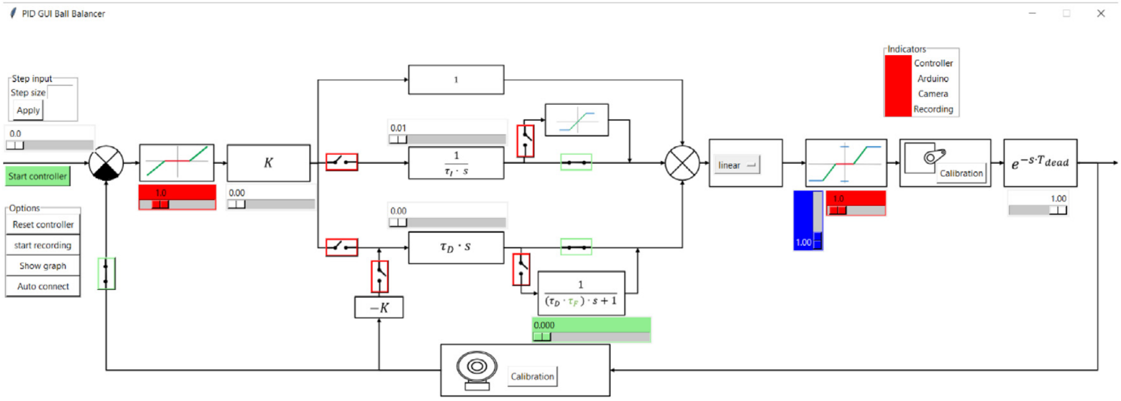
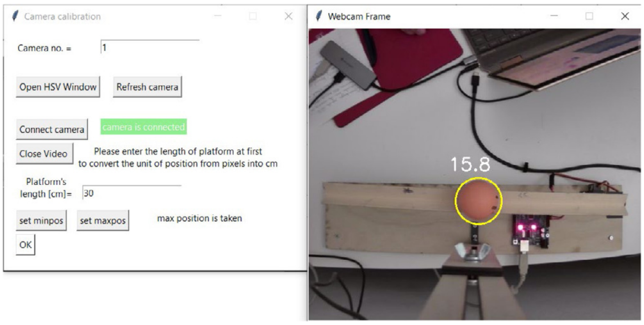


Fig. 1. The intuitive GUI that controls the physical setup. Different objects, such as sliders, buttons, and text boxes, are placed in appropriate places to give the user a clear sight of the whole control process.



Camera no. = 1

camera is connected

Please enter the length of platform at first
to convert the unit of position from pixels into cm

Platform's length [cm]= 30

max position is taken

Fig. 2. The GUI displaying the camera's detection of the ball and its position on the beam.

2.1. Graphical user interface (GUI)

Tkinter is a library used for GUI application development in Python [6]. The block diagram of the closed-loop system is used as the background for the main window, and different objects like sliders, buttons, and labels are placed in the appropriate places. This makes the GUI as structured and interactive as possible. Different buttons can access various windows, for instance, the camera's field of view, calibration windows for the camera, and the servo motor (see Fig. 1).

2.2. Acquiring the position of the ball using the camera

The software is compatible with any low-cost USB-connection camera to detect the ball on the beam. The open-source library, open CV2, is used in Python to track the ball by detecting the colour of the ball. The colour of the ball is represented in HSV (Hue, Saturation and Value) colour space which is widely used in computer vision and image analysis [7]. Hue represents the dominant colour observed through the eye. Saturation indicates the amount of white light mixed with Hue. The value represents the darkness of the colour the lower the value the darker the colour. Once the HSV colour range of the ball is known, the movement of the ball can be tracked based on this colour (see Fig. 2).

2.3. Serial communication with the Arduino

Pyserial library is used for serial communication for the Python programming language. The specified communication (COM) port connected to the microcontroller can be accessed with this library [8]. The purpose of communication between the computer and the Arduino is to send the calculated angle for servomotor. This angle should be converted into a byte array to send it to Arduino through a serial path. On the other hand, the Arduino is programmed in such a way that if there is data available on the serial port, it reads the data, converts it to integer, and sends it to the servo motor via the digital output pin, which can generate a PWM signal.

2.4. Implementation of the PID controller

The PID controller equation is typically represented in continuous-time or complex domain forms. They cannot be implemented in the software as it is. The transfer function that is represented in the s-domain should be transferred into the discrete-time domain using a suitable conversing technique. The ideal approach for this transformation is the use of the bi-stable formula, also known as the Tustin formula [9].

For the proportional controller, the Tustin formula cannot be implemented since there is not any s variable present, thus the discrete-time equation becomes: $P[n] = K_C \cdot E[n]$.

Applying the Tustin formula to the Laplace domain representation of the integral controller, the discrete time domain equation becomes: $I[n] = \frac{K_C}{\tau_I} \cdot \frac{\Delta T}{2} \cdot (E[n] + E\,[n-1]) + I\,[n+1]$ [10].

Applying the Tustin formula to the Laplace domain representation of the derivative controller, the discrete time domain equation becomes: $D[n] = \frac{2 \cdot K_C \cdot \tau_D}{\Delta T} \cdot (E[n] - E[n-1]) - D[n-1]$ [10].

2.5. Demonstration of various practical aspects

Besides the implementation of the PID controller, various practical aspects have also been implemented in the software that are linked with the imperfections of the components. For instance, the ideal derivative controller is not compatible with the real-life measurements because the high-frequency component (noise) present in the measurement gets amplified by the derivative controller [9]. Thus, there is a low-pass filter implemented with the derivative controller. Similarly, the appropriate technique to deal with the integral windup [11] and the derivative kick has also been implemented in the software [12].

3. Impacts

The developed software and the low-cost physical demonstration setup are being used as one of the practical sessions for the students taking the automation course within the faculty of engineering sciences at Vrije Universiteit Brussel (VUB). Though the demonstration setup is low in cost, it mimics all the primary problems that one can expect during an industrial PID control. This appears to be an effective way to reinforce the theoretical knowledge acquired in the lectures with practical experience. An article has been submitted and accepted by 17th IFAC Conference on Programmable Devices and Embedded Systems which briefly discuss the purpose and the user experience of the software. The article has been written to present the content that the user can learn from this software based on the experience of the user using this software. [13] In order to make this software and the physical setup available to the rest of the world, all the necessary materials have been published to the GitHub project that is required to replicate the physical setup and compile the software successfully.

In order to use the software easily, a self-written user manual is available on the teaching website sysidguy.eu. Some videos have been created to guide the user while using the software. Those videos are available on YouTube: Ball Balancing Beam.

4. Future developments

The authors plan to update the software according to the feedback of the users as this software is used by the undergraduate students at Vrije Universiteit Brussel. Further, the software will also be developed in C# or C++ programming language. As this software has been developed from scratch applying the fundamental of control theory, the core can be applied by other theses and projects. The equations described in paragraph 2.4 are the fundamental equations of the PID controller represented in the discrete domain. This can help the developer to understand the proper way of implementing the PID controller in the software for any application. Further, this also makes the software developer independent of the various PID packages available across different programming languages.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] B. Beauregard, 2018. Retrieved 12 28, 2021, from Arduino Playground-PIDLibrary.: https://playground.arduino.cc/Code/PIDLibrary/.

[2] M. Blanco, P. Csurcsia, B. Peeters, K. Janssens, W. Desmet, Nonlinearity assessment of MIMO electroacoustic systems on direct field environmental acoustic testing, in: Proceedings of ISMA 2018 - International Conference on Noise and Vibration Engineering and USD 2018 - International Conference on Uncertainty in Structural Dynamics. Leuven, Belgium, 2018.

[3] P. Csurcsia, J. Schoukens, I. Kollár, Identification of time-varying systems using a two-dimensional B-spline algorithm, in: IEEE I2MTC - International Instrumentation and Measurement Technology Conference. Graz, Austria, 2012.

[4] L.F. Chaparro, in: L.F. Chaparro (Ed.), Signals and Systems using MATLAB, Academic Press, Boston, ISBN: 9780123747167, 2011.

[5] G.F. Franklin, M.L. Workman, J.D. Powell, Digital Control of Dynamic Systems, Ellis-Kagle Press, Half Moon Bay, Calif, ISBN: 9780201119381, 2006.

[6] Tkinter, Introduction to {gui} programming with tkinter, 2013, Retrieved 11 30, 2021, from https://python-textbok.readthedocs.io/en/1.0/Introduction_to_GUI_Programming.html.

[7] A. Rosebrock, Ball tracking with openCV, 2015, Retrieved 10 27, 2021, from https://www.pyimagesearch.com/2015/09/14/ball-tracking-with-opencv/.

[8] PySerial, Welcome to pySerial's documentation, 2020, Retrieved 11 30, 2021, from https://pyserial.readthedocs.io/en/latest/.

[9] K.J. Aström, T. Hägglund, PID Controllers Theory Design and Tuning, Research Triangle Park, N.C. International Society for Measurement and Control, North Carolina, ISBN: 9781556175169, 1995.

[10] B. Pujan, Development of ball balancing demonstration setup, 2022, Brussels: Vrije Universiteit Brussel. Retrieved 02 12, 2022.

[11] S. Sen, A.K. Mukhopadhyay, Industrial Instrumentation, Control and Automation, Jaico Publishing House, ISBN: 9788184954098, 2013.

[12] Aström K.J. Luenberger, B. Wittenmark, Computer Controlled Systems, Theory and Design, Prentice-Hall, London, ISBN: 9780131643192, 1990.

[13] P. Csurcsia, P. Bhandari, D. Troyer, Development of a low-cost PID setup for engineering technology students, in: 17th IFAC Conference on Programmable Devices and Embedded Systems, 2022.