ControlDroid: A m-learning platform to learn and teach control systems in technology and engineering

Jonathan Álvarez Ariza
Dept. of Technology in Electronics
Corporación Universitaria Minuto de Dios
(UNIMINUTO), Bogotá, Colombia
jalvarez@uniminuto.edu

Hannsel Ricardo Neira Mercado
Dept. of Technology in Electronics
Corporación Universitaria Minuto de Dios
(UNIMINUTO), Bogotá, Colombia
hneiramerca@uniminuto.edu.co

***Abstract*—In this paper, we describe the m-learning platform *ControlDroid*, which allows the students to design and implement SISO control systems with their smartphones or tablets. The platform is composed by an application in Android and a development board to connect sensors and actuators for the control system. Regarding the application, the digital classic controllers (P, PI, PID) are built using the Google visual Tool *Blockly* that simplifies their implementation, focusing the attention of the students in the aspects of plant identification, simulation and design, instead of the algorithms needed in the implementation of a controller. To test the platform, we perform an experiment to control the velocity ω of the DC motor (EMG30) and we expose the respective results in the manuscript. The concept of the platform represents a curricular transformation in the area of control systems, taking the advantage of the mobile devices and offering the possibility to the students learn control systems in their homes without an expensive infrastructure, aspect that takes more relevance given the current challenges of the education in the COVID-19 pandemic.**

***Index Terms*—Control systems, m-learning, engineering education, algorithm visualizations, digital controllers.**

I. Introduction

Control systems are usually a complex field in engineering and technology because the students must have a set of cognitive skills in math and physics in aspects such as simulating, modelling, identification and designing that could result overwhelming in some cases or even generate university dropout [1]. In addition, control systems gather computing, signal conditioning and instrumentation. The matter becomes more complex whether we consider the high costs of the needed laboratory equipment for control systems, e.g., control plants, actuators, or controllers. Thus, students must move to the laboratories in their institutions to access to this equipment. Aside these factors, students enrolled in engineering and technology programs claim for a renovation of the traditional curricula towards courses that considering, for example, the use of technologies and educational methodologies that result more inclusive and allow to enhance their learning. Also, the COVID-19 pandemic has represented a challenge to create and accelerate the incorporation of these methodologies and technologies in higher education.

In order to contribute with proposals that address these matters, this investigation describes the platform *ControlDroid* that has been designed to teach and learn Single-Input and Single-Output (SISO) systems in introductory courses of automatic control or control systems. The platform has an Android application in which the students can build different Algorithm Visualizations (AVs) through graphical blocks (*Blocky*) [2] to perform tasks as plant identification and implementation of classic controllers such as Proportional (P), Proportional-Integral (PI) and Proportional-Integral-Derivative (PID). Also, data of the experiments can be exported in CVS format to further processing in software as MATLAB or SCILAB. Once a student completes his/her control algorithm, this can be downloaded to a development board that employs a micro-controller ATMEGA328P, using the Bluetooth Protocol with the feature Serial Port Profile (SPP). We have chosen the Bluetooth Protocol to prevent access problems to the platform due to drivers' compatibility in different versions of Android. For instance, some smartphones or tablets do not contain the feature USB OTG to connect devices, instead, Bluetooth is a concurrent technology between several Android versions, which makes more accessible the platform to the students.

From an educational point of view, the platform is a proposal that takes into account the m-learning [3], [4] in where learning and learners are mobile, and the students can draw on another spaces of learning in their homes or jobs without excessive costs in the infrastructure. Regarding this last point, some initiatives are arisen, providing alternatives to the students so as to explore and learn concepts in Control Systems such as plant identification, controller design, feedback or stability, some of them employing (AVs) and MATLAB [5], [6], [7]. So that, our interest is to complement these works with a m-learning platform that will be used in the courses of control systems in our institution given the current situation of the higher education under the COVID-19 pandemic.

According to the aforementioned, this work is divided as follows: Section II describes the platform to technical level in the components of hardware and software. Section III shows an experiment to control the velocity ω of the DC motor (EMG30) [8] carried-out with the platform. Finally, section IV and V outlines the discussion, conclusions and further work of this proposal.

II. Platform description

A. Software and hardware components

The software interface of *ControlDroid* is an App compatible from Android 5.0 until android 10.0. This application

978-1-7281-7118-0/20/$31.00 ©2020 IEEE

is composed by two layers, namely, User Interface (UI) and Application Framework that are depicted in Fig.(1).

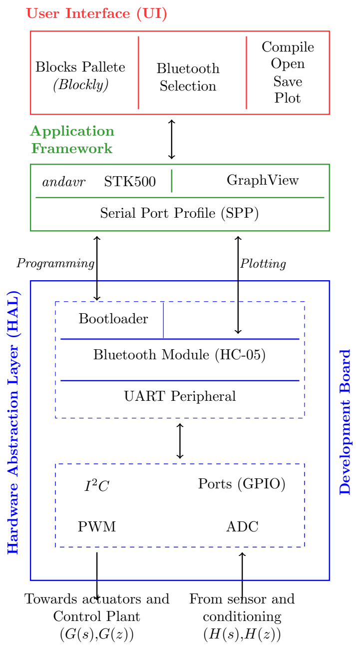

Fig. 1. Software and hardware structure of *ControlDroid*.

The UI employs *Blockly* for the different graphical blocks that are distributed by categories as it is depicted in the Fig.(2). In addition, we have built a *control* category for the classic digital controllers (P, PI, PID) that have different parameters as set point, process scale, the constants (K_p, K_i, K_d), depending on the type controller and the sensor input ($H(s)$, $H(z)$). These elements allow the students to configure the type of classic controller according to the design parameters for the control plant ($G(s), G(z)$).

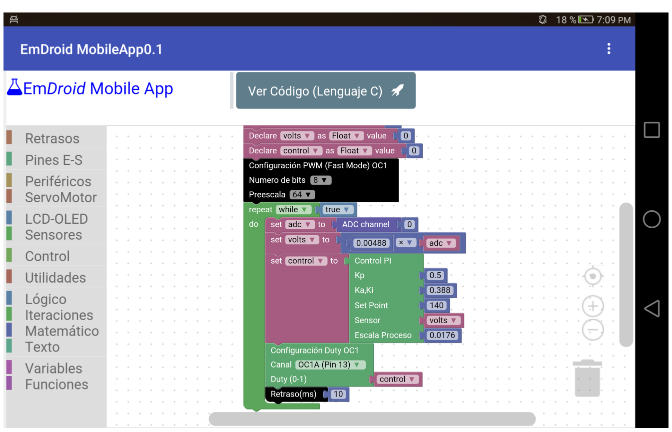

Fig. 2. User Interface of *ControlDroid*. *Left*. Blocks palette. *Center*. Working area. *Top*. User menu and code visualization button.

To meet with the previous requirements, the core of *Blockly* was modified, adding new additional blocks for different functions such as port configuration, Pulse Width Modulation (PWM) or Analog to Digital Converter (ADC). Each block was transformed in C language and tested in the used microcontroller in the development board (ATMEGA328p), complying with the requirements of the compiler (AVR-GCC v.4.9) in its Android version known as *andavr* [9].

When a student creates his/her control algorithm selects in the application the Bluetooth name, afterwards, the algorithm is compiled and downloaded through the feature Serial Port Profile (SPP) that emulates a serial port to transfer the code compiled (*.hex* file) towards the microcontroller, which contains a bootloader to detect a new programming request from user. The application monitors the transaction of the code using the protocol STK500.

A parallel function in the application is to plot. The students can plot the results of their designed controllers, creating a visual algorithm to send the information, employing the peripheral UART. The data are transferred to the Android library *GraphView* that allows to plot the data in real-time. Data are saved in the folder of the application in .CVS format.

Regarding the development board, it contains a microcontroller ATMEGA328P. The board is equipped with Arduino connectors to the students plug in the sensors and actuators according to their needs and several pins to supply 5V and 3.3V. Moreover, the board has a Bluetooth Module (HC-05) to program the microcontroller and transfer data of the control process to plot in the application.

III. Control Systems Experiments

In this section, it is shown an experiment to control de velocity ω of the proposed control plant (DC motor EMG30). Also, we followed the steps that typically one student performs concerning the controller design and the usage of the platform.

A. Identification

In this stage, the student identifies the plant, applying a step source that can change along the time. Although a DC motor is a second order plant, for this motor, a first order model is an accurate approximation to simplify the controller design. Therefore, the student must recognize the parameters of K and τ according to the expression (1) in the Laplace domain (s).

$$G(s) = \frac{K}{\tau s + 1} \tag{1}$$

To identify these parameters, the student can build the following algorithm that changes the duty cycle (k) of a PWM signal with $f \approx 1KHz$.

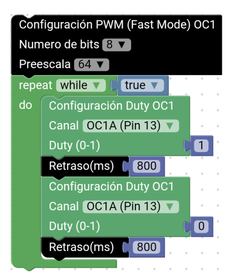

Fig. 3. Visual Algorithm to control the PWM peripheral in the process of plant identification.

The EMG30 motor contains a quadrature encoder that produces 360 pulses per revolution with a rated speed of 170 rpm to 12V according to the specifications provided by the manufacturer. We designed a Printed Circuit Board (PCB) with the linear frequency to voltage converter LM331, which is interfaced with the development board as it is illustrated in Fig.(4). The gain of this device with its conditioning for the experiment considering the frequency of the PWM $f = 1KHz$ is $H(s) = 0.01496$.

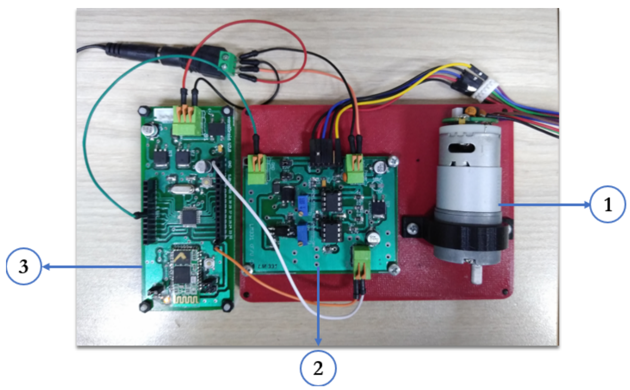

Fig. 4. Low-cost control system experiment setup. 1. EMG30 Motor, 2. LM331 board, 3. *ControlDroid* board.

Because the major part of the students do not have an oscilloscope in their homes, the application has a plotter to solve this problem. The plotter captures the data through the 10-bit Analog to Digital Converter (ADC) peripheral of the microcontroller with a preconfigured sample time of $T_s = 10ms$, that can be changed by the student directly in the menu of the application with a minimum of $T_s = 1ms$ and a processing time by the plotter of 2.5ms. An example of the graph generated by the algorithm of Fig.(3) and saved in .CVS format is illustrated as follows:

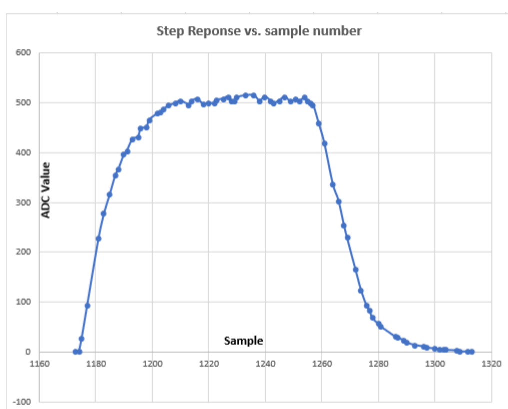

Fig. 5. Control Plant $G(s)$ step response for the algorithm of Fig.(3) produced by the plotter in .CVS format. $V_{motor} = 12V$.

When the DC motor has a maximum velocity (170 rpm), the LM331 board provides a voltage of 2.48V (ADC=508) and 0 rpm correspond to 0V (ADC=0). The student can recognize the parameters of the plant as $\tau = 0.13sec$ and $K = 170$, approximately.

B. Controller design

There are several techniques to design a controller (P, PI, PID) as root locus, pole location, space state, analytical, etc., that are taught in control systems. In this paper, we chose the analytical technique and the PI controller, but professors or instructors can select the method and controller that consider adequate to experiment with the platform. In function of the identified parameters K and τ, we want a desired plant, starting in the Laplace domain (s) with a settling time $(t_s = 0.5sec)$ and a Percentage Overshoot (PO) of 2%. The desired plant $G_d(s)$ is described by Eq.(2) with $t_s = \frac{4}{\zeta\omega_n}$ and the $PO = 100 \cdot e^{-\frac{\zeta\omega_n}{\sqrt{1-\zeta^2}}}$. These equations yield to $\zeta = 0.7798$ and $\omega_n = 10.26\frac{rad}{sec}$.

$$G_d(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} = \frac{105.3}{s^2 + 16s + 105.3} \qquad (2)$$

To design the controller, we followed the SISO control structure shown in Fig.(6).

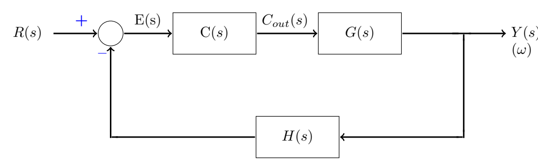

Fig. 6. Control block diagram for the experiment. Control Plant $G(s)$, controller $C(s)$, sensor and conditioning $H(s)$, reference $R(s)$, control output process $Y(s)$.

The reduction of this control system is $\frac{Y(s)}{R(s)} = \frac{C(s)\cdot G(s)}{1+C(s)\cdot H(s)}$. The PI control in the Laplace domain can be expressed as $\frac{C_{out}(s)}{E(s)} = \frac{K_P(s+a)}{s}$ where $a = \frac{K_i}{K_P}$, $E(s)$ is the error and K_i y K_p are the integral and proportional constants, respectively. In the analytical method, we want to approximate the poles of the control system reduction to the desired plant in (2), in this way, replacing the previous expressions in the mentioned reduction, we have:

$$\frac{Y(s)}{R(s)} = \frac{1308 \cdot K_p(s+a)}{s^2 + 7.692 \cdot (1 + 2.5435 \cdot K_p) + 19.57 \cdot K_p \cdot a} \qquad (3)$$

The expression (3) yields to the values $K_p = 0.4345, K_i = 5.52$. Due to that the digital controllers in the application are implemented through *difference equations*, the PI controller in Laplace domain must be converted in Z domain, using a method such as Zero-Order Hold (ZOH) or bilinear transformation (Tustin). These methods can be taught to the students in MATLAB or SCILAB, e.g., employing the command *c2d*. Thus, the PI controller $\frac{C_{out}(s)}{E(s)} = \frac{0.4345(s+12.7)}{s}$ can be mapped in Z domain using the method ZOH and a sampling time $T_s = \frac{1}{10} \cdot \tau = 0.013sec$, given as a result $\frac{C_{out}(z)}{E(z)} = \frac{0.436z-0.363}{z-1}$. Transforming the previous expression to a difference equation, yield to $c_{out}(kT) = 0.436 \cdot e(kT) - 0.363 \cdot e(kT-1) + c_{out}(kT-1)$, where $c_{out}(kT)$ is the output of the controller and $e(kT)$ is the error. This type of equation is implemented for the PI controller and a similar structure for the PID controller.

C. Controller implementation

To implement the PI controller, the student uses the block *Control PI* available in the category *Control* of the application blocks. The algorithm has the following structure:

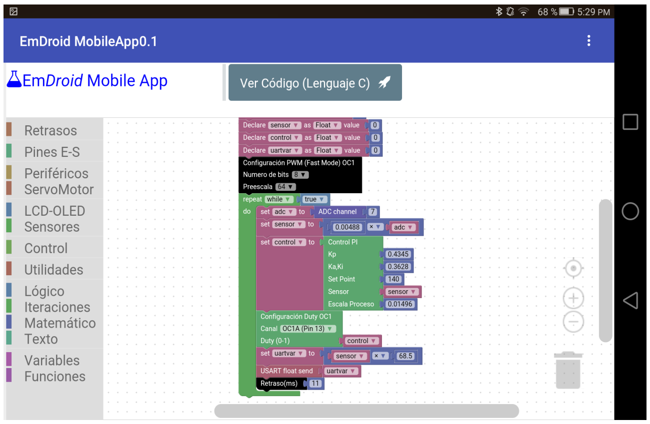

Fig. 7. Control algorithm for the PI Controller, $K_p = 0.4345, K_i = 5.52, a = 0.3628$, Setpoint (140 rpm), ADC channel (7).

The difference equation needs the value of K_p and a to implement the controller. Additionally, the student must set up some parameters such as setpoint, sensor and the process scale that adjusts the reference in rpm to the value of voltage to be processed by the controller. The sensor in this case is composed by the encoder, the LM331 and its conditioning, which is reading by the ADC in the channel (0 to 7) that the student selects. Then, the controller output is sent to the block OC1 that changes the duty cycle of the PWM signal for the DC motor as described.

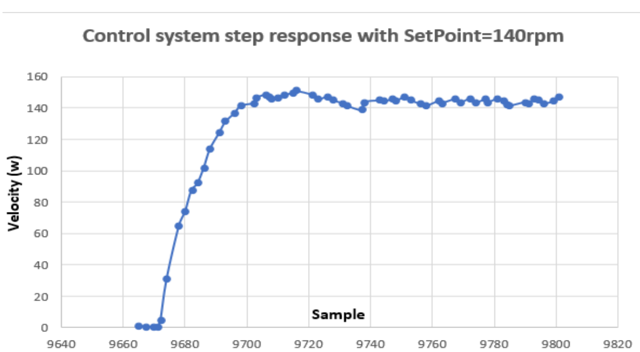

Fig. 8. Control system step response provided by the plotter with the PI controller. Setpoint (140 rpm).

IV. Discussion

In the development of the control system, we followed the steps that one student can perform in control systems. However, the platform is flexible to allow different types of approaches to design a controller because it helps with the steps of plant identification and controller implementation, focusing the attention of the students in the analysis of the control system. The platform uses graphical blocks that serve to build a Visual Algorithm (VA), which is converted to C language for the microcontroller. If the student wants to see the respective code, he/she can press the button *see code (C language)*. The flexibility of the platform allows the students to use their tables or smartphones to understand the concepts of control systems. Furthermore, the low-cost kit with the elements of the Table.(I) can help to the students to learn control systems without an expensive infrastructure. Information about the platform *ControlDroid* can be found in [10].

TABLE I
Costs of the control systems kit.

Component	Quantity	Cost (USD)
EMG30 (DC motor)	1	38
LM331 board	1	15
ControlDroid development board	1	25
DC adapter (12V, 1.5A)	1	4
Wires	10	2
Total ($)	-	**84**

V. Conclusions

In this work, we described the platform *ControlDroid* which is focused on the students learn control systems using their smartphones or tablets. The concept of the platform represents a curricular reflection, considering the current context of the COVID-19 pandemic, the technology and the implications that they can have in engineering and technology education, specifically, in the curricula of Control Systems. With the proposed low-cost kit and the platform students can perform different experiments in their homes to learn the concepts of control systems without an expensive infrastructure and equipment. Further work will consist in validate and assess the educational implications of the platform usage in the learning of the students. Also, new graphical blocks to extend the functionalities and types of the controllers will be added.

References

[1] O. I. Higuera Martínez *et al.*, "Deserción estudiantil en colombia y los programas de ingeniería de la uptc seccional sogamoso," 2017.

[2] E. Pasternak, R. Fenichel, and A. N. Marshall, "Tips for creating a block language with blockly," in *2017 IEEE Blocks and Beyond Workshop (B&B)*. IEEE, 2017, pp. 21–24.

[3] M. Sarrab, L. Elgamel, and H. Aldabbas, "Mobile learning (m-learning) and educational environments," *International journal of distributed and parallel systems*, vol. 3, no. 4, p. 31, 2012.

[4] J. Traxler, "Defining mobile learning," in *IADIS International Conference Mobile Learning*, 2005, pp. 261–266.

[5] V. Viswanatha and R. V. S. Reddy, "Digital control of buck converter using arduino microcontroller for low power applications," in *2017 International Conference On Smart Technologies For Smart Nation (SmartTechCon)*. IEEE, 2017, pp. 439–443.

[6] R. M. Reck and R. S. Sreenivas, "Developing an affordable and portable control systems laboratory kit with a raspberry pi," *Electronics*, vol. 5, no. 3, p. 36, 2016.

[7] D. Weintrop, D. C. Shepherd, P. Francis, and D. Franklin, "Blockly goes to work: Block-based programming for industrial robots," in *2017 IEEE Blocks and Beyond Workshop (B&B)*. IEEE, 2017, pp. 29–36.

[8] Robot-Electronics. (2020) EMG30 motor specifications. [Online]. Available: https://www.robot-electronics.co.uk/htm/emg30.htm

[9] (2020) Android AVR compiler specifications. [Online]. Available: https://code.google.com/archive/p/andavr/

[10] (2020) ControlDroid specifications. [Online]. Available: http://www.seconlearning.com/WikiDokuControlDroid/doku.php?id=inicio

[11] D. Galán, L. de la Torre, S. Dormido, R. Heradio, and F. Esquembre, "Blockly experiments for ejss laboratories," in *2017 4th Experiment@ International Conference (exp. at'17)*. IEEE, 2017, pp. 139–140.