2019 12th Asian Control Conference (ASCC)
Kitakyusyu International Conference Center
Japan, June 9-12, 2019

OptoShield: A Low-Cost Tool for Control and Mechatronics Education

Gergely Takács*, Tibor Konkoly, Martin Gulan
Institute of Automation, Measurement and Applied Informatics
Faculty of Mechanical Engineering
Slovak University of Technology in Bratislava
Nám Slobody 17, 812 31 Bratislava, Slovakia
* Corresponding author: gergely.takacs@stuba.sk

***Abstract*—Commercial educational tools for teaching feedback concepts in control systems engineering and mechatronics are generally expensive, large, complex and sensitive instruments; therefore cannot be used for homework assignments outside the classroom. This paper presents a novel open-source teaching aid that can be readily manufactured under €3 and as such, is financially accessible to students and universities operating on a tight budget. The device proposed in this paper implements a simple optical experiment, allowing students to follow a prescribed brightness level using feedback control concepts. In addition to real-time control, the apparatus is also suitable for teaching signal processing and system identification principles. The educational aid presented here is attached to the popular Arduino microcontroller prototyping platform in a form of a miniature extension shield, thereby using a standardized physical layout and software ecosystem. We describe the hardware, the application programming interface (API) and library written in C/C++ for the Arduino integrated development environment (IDE) and the accompanying Simulink toolbox; moreover present sample exercises for system identification and proportional-integral-derivative (PID) control.**

***Index Terms*—control engineering education, optical control, student experiments, feedback circuits, microcontrollers, Arduino, low-cost**

I. Introduction

The education of professionals working in the field of control engineering and mechatronics must include two key components: solid fundamentals in the theoretical aspects of their future trade and practical hands-on experience with real hardware implementation. The latter requires laboratory experiments that are traditionally carried out on purpose-built commercial engineering education tools.

Numerous vendors offer their products to academic institutions; unfortunately most of these devices come at a steep price. These products, such as linear and rotary inverted pendulums, ball-on-plate systems, magnetic levitation or simplified helicopters provide an excellent introduction to real control problems. However, for most universities thousands or even tens of thousands of Euros invested in one workstation is financially prohibitive, in particular when in reality 5–10 of them are needed for a single class. This is not the only issue with currently available laboratory hardware, as these devices are usually paired with a closed-source software bundle that is yet again linked to commercial software such as MATLAB or LabView. Furthermore, students can only experiment in the laboratory and under close supervision; they may not take equipment home to continue working on their assignments. This, of course, is because of the high price, large size and delicate nature of these devices.

Fig. 1. The OptoShield control systems engineering educational aid mounted to an Arduino Uno.

In the past couple years we have been witnessing a revolution in teaching microcontroller and embedded software engineering thanks to the microcontroller unit (MCU) prototyping boards known collectively under the 'Arduino' name. The secret to the success of the Arduino boards is standardization. The Arduino Uno in particular became the gold-standard for education, since the hardware layout and functionality is unified, the software is free and well maintained and the support and community of helpful

The authors gratefully acknowledge the contribution of the Slovak Research and Development Agency (APVV) under the contract APVV-14-0399. The authors appreciate the financial support provided by the Cultural and Educational Grant Agency (KEGA) under the contract 005STU-4/2018.

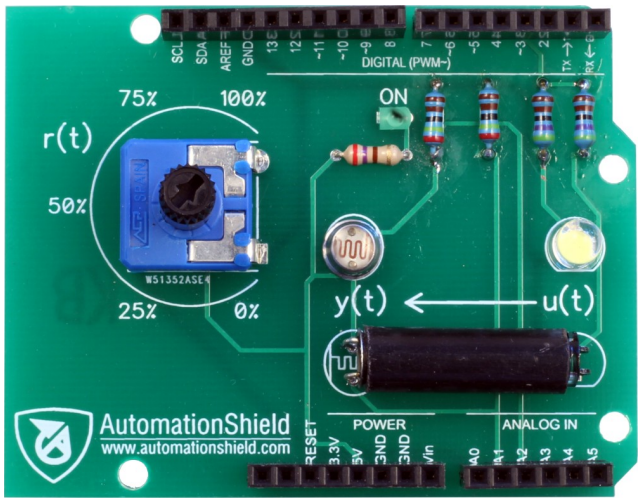

Fig. 2. Top view of the device showing its essential components.

TABLE I
BILL OF MATERIALS AND PRICE ESTIMATES FOR A SINGLE DEVICE, EXCLUDING LABOR. ALL PRICES SHOWN FOR MINIMUM ORDERS IN EUROS.

Part	Designator	Pcs.	Price	Sum
LDR, 5 mm, 5–10 kΩ	LDR1,LDR2	2	0.65	1.3
Resistor, 2.4 kΩ	R1, R2	2	0.0074	0.0148
Resistor, 4.7 kΩ	R3, R4	2	0.0074	0.0148
LED 5 mm, clear	D1, D2	2	0.116	0.232
Potentiometer, 10 kΩ	POT1	1	0.2760	0.2760
Potentiometer knob	—	1	0.0920	0.0920
Header 6 pin, F	—	1	0.085	0.085
Header 8 pin, F	—	2	0.096	0.192
Header 10 pin, F	—	1	0.096	0.096
Tube, Φ=5 mm	—	20 mm	0.10	0.10
PCB	—	1	0.50	0.50
Total				€2.9

enthusiasts is especially active. The functionality of basic boards can be readily extended since the physical and electrical layout of the Uno board known as 'R3' is retained across numerous other products. As detailed by Garrigós et al. [1], extra hardware known as 'Shields' can be simply pushed onto the board through the header pins extending the base board functionality for classroom use. The Arduino has made a lasting appearance in the education of feedback control, system identification and signal processing concepts as well. The effect of Arduino-based control projects on student performance and satisfaction has been also quantified, demonstrating positive learning outcome and student satisfaction [2].

While there are numerous engineering projects based on the Arduino for digital signal processing [3], electric engineering [1] and robotics [4]; there are fewer examples incorporating feedback-control concepts. Uyanik and Catalbas propose a motor-control experiment where the power electronics are implemented by the Arduino Motor Shield [5]. The well-known inverted pendulum on a cart is re-imagined as an open-source Arduino-based device in [6], whereas the familiar ball-on-plate experiment has also been transformed into a relatively affordable Arduino variant [7]. The total price of these devices usually ranges from a hundred to few hundred Euros and require custom-manufactured or improvised mechanical parts, base units or support enclosures. Thus, most of Arduino-based control education tools introduced in the literature are one-off purpose-built devices.

A beam-balancing system actuated by a fan was presented in [8]. A similar concept has been recently introduced by Kalúz et al. as well [9], albeit in a more refined form that is suitable for mass-production to a greater extent. Their open-source Arduino-based experimental device implements a combination of a fan and a moving flap with indirect position feedback called 'Flexy'. The authors of this paper employed this device for an entire semester with two groups of 14 students in a subject aimed at teaching control engineering concepts applied on practical microcontroller technology. While the visual feedback of the device proves to be a valuable teaching asset, the construction of the device requires access to a computer-numerical controlled (CNC) laser-cutting machine to prepare its enclosure and mechanical parts, encompasses the Arduino itself, contains a non-standardised improvised prototype board for its power its electronics [10] and its acquisition price was €120 a piece. Even though the participating students expressed keen interest in buying or constructing their own Flexy devices, the hardware construction and projected price proved to be prohibitive in practice.

This paper presents a novel open-source didactic tool to teach basic concepts in control engineering and mechatronics. Unlike in the case of other didactic devices described in the literature; our proposal is founded on the idea that the hardware should be integrated on a Shield that fits on a single printed circuit board (PCB) copying the now standardized R3 header pinout of the Arduino ecosystem. This way, the device can be freely attached and removed to and from different Arduino models, thus dramatically reducing unit cost.

Here we introduce a single input, single output experimental device that is compatible with most Arduino prototyping boards and can be manufactured under €3. The hardware is open-source, with CAD and manufacturing files ready for production. The proposed device shall be referred to as 'OptoShield' (see Fig. 1) in the upcoming discussion. It implements an optical feedback loop that can be utilized in teaching various design concepts in control theory, system identification or digital signal processing.

II. HARDWARE DESCRIPTION

The feedback path consists of a light emitting diode (LED) in the role of the actuator and a light dependent resistor (LDR) as a sensor. The components are encased in an opaque tube to prevent the much brighter daylight to interfere with the experiments. As the device is aimed at beginners, there is an external LED and LDR included as well. Because neither of these components are accessible or visible, students may use the external counterparts for experimentation and testing. Finally, there is a potentiometer that can be programmed to perform any role, albeit its main function is to provide the user-adjustable

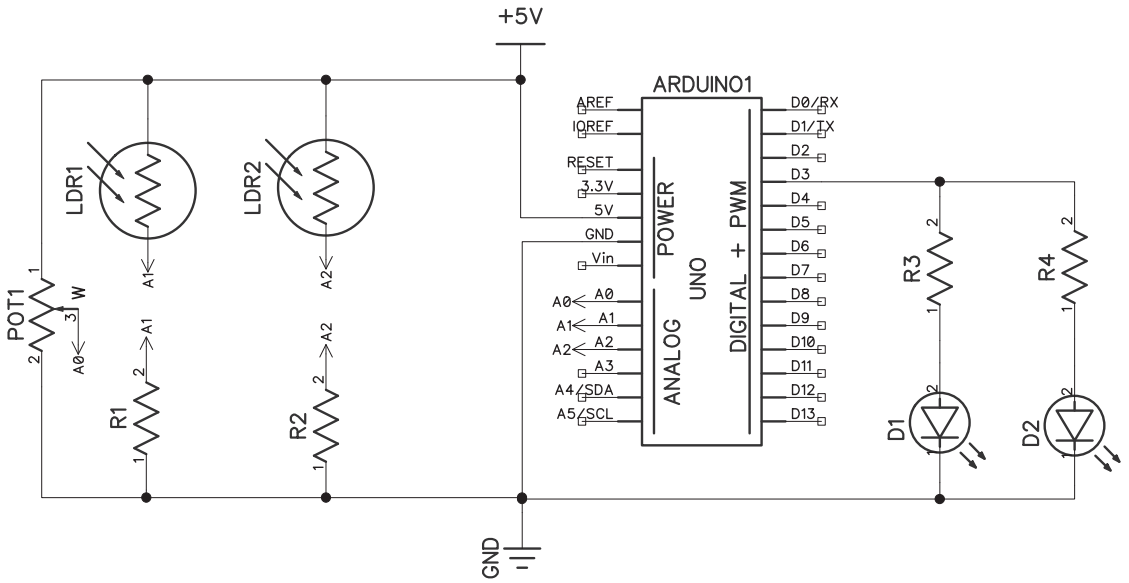

Fig. 3. Electrical schematic of the OptoShield excluding the power indicator LED.

reference brightness to the feedback loop. Since the built-in power indicator LED of the Arduino board is covered by the OptoShield, an additional LED signals that the device is powered on. All of these components are shown on a fully assembled board in Fig. 2.

The electric schematic of the OptoShield is shown in Fig. 3. The current to both the main LED (D1) and its external duplicate (D2) is limited by series resistors R3 and R4. The value of these resistors has been chosen so as to create a maximum brightness that is within the sensing range of the LDR. Both LED are connected to digital output D3 of the Arduino, therefore the brightness of the external LED copies that of the main one.

The main LDR acting as a sensor (LDR1) is connected to the A1 analog input of the Arduino through a voltage divider configuration using resistor R1. Its value has been chosen to increase the resolution at the output measurement when combined with the brightness of the LED. This is repeated for the external sensor with LDR2 and R2 connected to the input A2. Finally, the potentiometer is connected to the A0 input of the Arduino, while the power LED with its series resistor is omitted from the scheme.

Table I lists the parts and the components of OptoShield, their quantity required to manufacture a single board and estimated price. The price estimates are given in Euros for small quantity production, minimum orders and exclude labor price. As it is clear from the table, the total comes to less than €3 a piece, which in high production volumes makes the OptoShield especially affordable. Components have been purposefully selected in through-hole packages, making the final assembly possible for even those less experienced with a soldering iron.

The double-layer printed circuit board (PCB) copies the shape of the Arduino Uno, while the pins conform to the R3 standardized layout. The use of stackable headers enables the combination of the shield with other components or devices, e.g. for data logging or Ethernet. The circuit board illustrated in Fig. 4 was designed in the free version

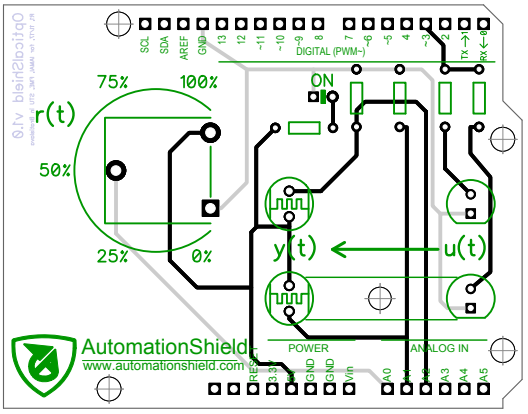

Fig. 4. The PCB of the OptoShield showing all layers (1:1 scale).

of the DipTrace schematic and PCB design software, with all layered files including the layouts ready for production are accessible in [11]. The silkscreen of the PCB designates the actuator as input ($u(t)$), the sensor as output ($y(t)$) and the potentiometer as reference ($r(t)$).

III. APPLICATION PROGRAMMERS' INTERFACE

Although creating the essential software functionality to initialize and calibrate the device, read the sensor and operate the actuator may be the inherent part of the learning process, an application programming interface (API) is also available in [11].

A. C/C++ API for Arduino IDE

An all encompassing learning curriculum should contain the microcontroller realization of feedback control, therefore the API has been written in the Arduino dialect of the C/C++ language. The API for the OptoShield is contained in the so-called AutomationShield library for the Arduino IDE. This library is an open-source hardware and

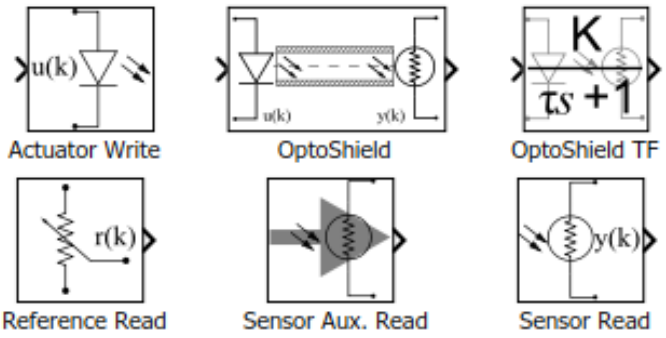

Fig. 5. Blocks available from the Simulink API.

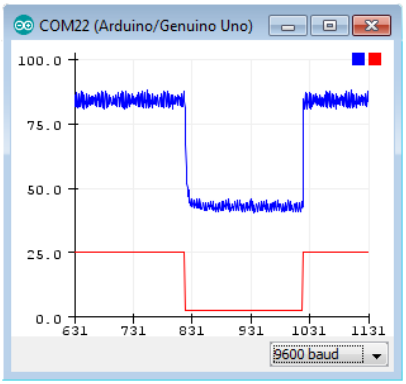

Fig. 6. Process input and output in the Serial Plotter.

software project, which in addition to the OptoShield, will implement other hardware similar to the present one.

The OptoShield API is located in the `OptoShield.h` header and a corresponding source file; these implement the `OptoClass` class which is then declared as default `OptoShield` instance. The direction of the inputs and outputs is initialized by calling

```
OptoShield.begin();
```

which must be followed by

```
OptoShield.calibrate();
```

to re-scale the input and output values. Because the LDR cannot measure physically valid units, the signal to the LED and from the LDR will be given in percents of the full scale. The method `calibrated()` thus finds the minimal and maximal analog-to-digital converter (ADC) levels measured at the sensor. The `returnCalibrated()` method returns a flag informing on the calibration state, while the `returnMinVal()` and `returnMaxVal()` method return the sensor calibration levels.

The sensor can be read by calling

```
y=OptoShield.sensorRead();
```

which returns a floating-point number in the calibrated range of 0–100% representing process output $y(k)$. The voltage at the main sensor is accessible through the `sensorReadVoltage()` method as well, while the auxiliary sensor can be accessed via `sensorAuxRead()`.

Both LED are activated at the same time through the

```
OptoShield.actuatorWrite(u);
```

method. This accepts input $u(k)$ as a floating-point number in the range of 0-100%, which is then sent to the pulse-width modulation (PWM) enabled port.

The onboard potentiometer can be used in any role, but the most straightforward one is to read a desired setpoint $r(k)$ using

```
r=OptoShield.referenceRead();
```

returning a floating-point number between 0-100%.

B. Simulink API

The capabilities of the Simulink Support Package for Arduino Hardware have lately vastly improved, allowing code generation and two-way communication between microcontroller and computer even for the Arduino Uno. This means that reference setpoints can be sent to the board directly from the Simulink scheme, while experiment results may be viewed and logged online.

The API for the OptoShield has been created and tested in the 2018b release of MATLAB and Simulink. The resulting library toolbox is illustrated in Fig. 5. The onboard LEDs have a dedicated block to which the user supplies a signal from the range of 0–100 (%). The LDR and its auxiliary twin are serviced by individual algorithmic blocks; in these the user may select the desired type of outputs such as voltage, ADC levels or a manually calibrated signal in percentages. The user may access the onboard potentiometer in a similar fashion.

The most important block in the Simulink API represents the inputs and outputs of the physical system in a single package. The block automatically calibrates the sensor to the available range, then accepts a saturated input signal in the range of 0–100 (%) and reads the calibrated brightness signal from the main LDR. The process dynamics is also represented by a continuous linear transfer function for simulation-only exercises.

IV. Demonstration experiments

The limited scope of this article does not enable us to present the full spectrum of didactic experiments possible with the proposed hardware. Here we will merely introduce the typical dynamic nature of the process and the possibilities offered by the proposed device.

A. System identification

The AutomationShield library for OptoShield incorporates several worked examples, one of these[1] illustrates the input-output step response of the onboard optical system. A more detailed example[2] runs through a preset array of open-loop inputs, incorporates an interrupt-based sampling system and lists the results to the serial communication line. The dynamic response of the system can be followed through the built-in Serial Plotter of the Arduino IDE. This enables one to display the process dynamics within the development environment itself, see Fig. 6.

[1]See `OptoShield_StepResponse.ino`.
[2]See `OptoShield_Identification.ino`.

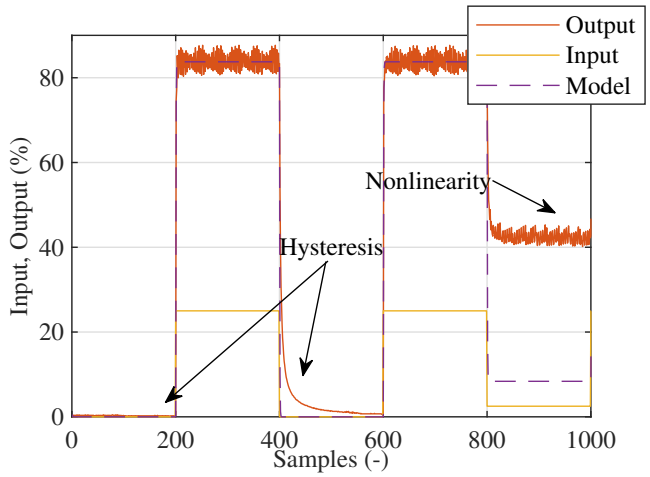

Fig. 7. Identification and modeling of the underlying process.

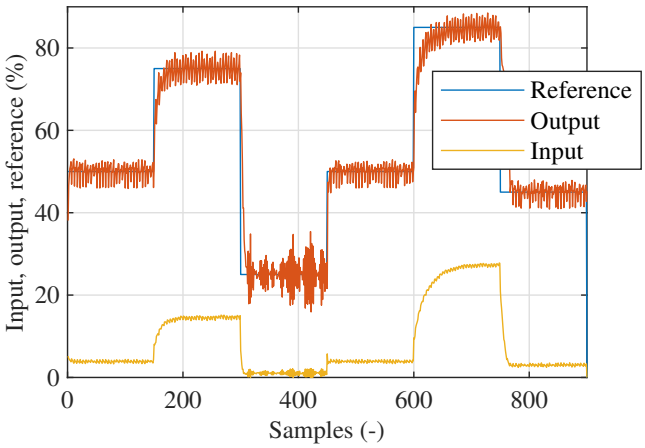

Fig. 8. Closed-loop PID control of the LED brightness.

Apart from this simple graphical representation, the library includes a function to read the results to the MATLAB workspace. This way students can perform system identification procedures on the collected data. Figure 7 shows the input and output data gathered through the latter example. Students may, for example, identify a continuous-time first-order process model using the MATLAB System Identification Toolbox and compare the model to the measurement results. As a result, the first-order transfer function

$$G(s) = \frac{3.35}{1 + 0.0041s} \tag{1}$$

with a K=3.35 (-) D.C. gain and T=0.0024 s time constant produces a 96.46% match with measurement data, as long as we compare a unit step from zero level up to the expected working range of the optical tunnel.

However, as it is clear from Fig. 7 students can also learn that this simplified dynamic representation is far from perfect, as the real process demonstrates hysteresis and significant nonlinearity. More advanced curriculum can then create models compensating for these effects as well. The scope of the current paper does not allow to explore the exact source of these nonlinearities in detail. Although most manufacturers regard the response of their LDR to lightning conditions fairly linear [12], [13], one possible source of the hysteresis under dynamic changes is the discrepancy between the rise and fall time, that can be an order of magnitude different for certain models [12], [13].

B. PID control

The gamut of feedback control algorithms that can be implemented using the proposed device are only limited by the performance of the microcontroller itself. Since all engineering curricula in feedback control explain proportional-integral-derivative (PID) control, we will demonstrate the closed loop behavior of our device using this particular approach.

C. Arduino IDE

The examples included within the library contain a PID control example with manual reference levels set through the potentiometer and an experiment with pre-determined setpoints. The example code initializes the board and by calling the generic `AutomationShield.h` header also makes use of the sampling and PID functionality of the library. Discrete sampling is realized by launching an interrupt-enabled callback routine at each sampling period. Students may be required to program their own PID routines, or alternatively, the library contains PID algorithms in both absolute and incremental forms with integral wind-up and saturation limits.

A PID controlled closed-loop experiment is shown in Fig. 8, where the algorithm was tuned to $K_\mathrm{P} = 0.1$, $T_\mathrm{I} = 0.015$ and T_D to a rather symbolic $T_\mathrm{D} = 0.015$. Despite the true hysteretic an nonlinear nature of the process the PID algorithm handles feedback control quite well. Note that the output oscillates around the setpoint—especially at lower reference levels—because the LED is powered by a PWM signal instead of a stable true analog input from a digital-to-analog converter (DAC).

D. Simulink

The Simulink API introduced in Sect. III-B allows students to implement system identification and feedback control experiments quickly and intuitively. One can create controllers from scratch or use the great variety of algorithmic blocks that come with Simulink. The schemes are compiled to AVR-compatible machine code, while two-way communication is preserved between the block scheme and computer. This way students may use switches or interactive tuning parameters in their design.

Here we will illustrate the possibilities of the hardware and the API by a simple PID controller that is shown in Fig. 9. The feedback loop uses a tracking reference given by a slider on the computer screen, while the control moves are computed by the built-in discrete PID block. Results are transferred back to the computer and can be viewed and logged in a Scope block (see Fig. 10).

V. Conclusion

A novel hardware device that implements an optical feedback control experiment for an extremely low price

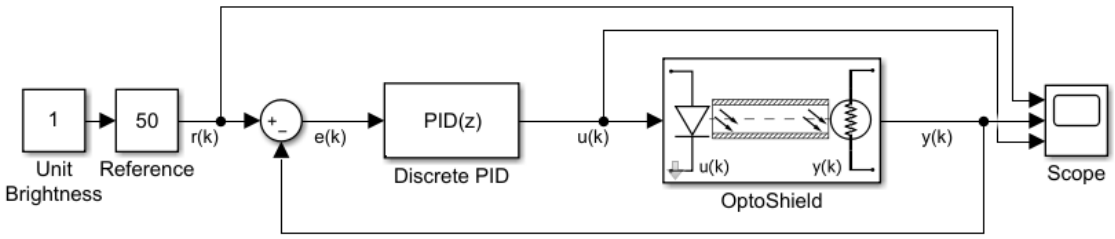

Fig. 9. PID control feedback loop of the proposed device in Simulink.

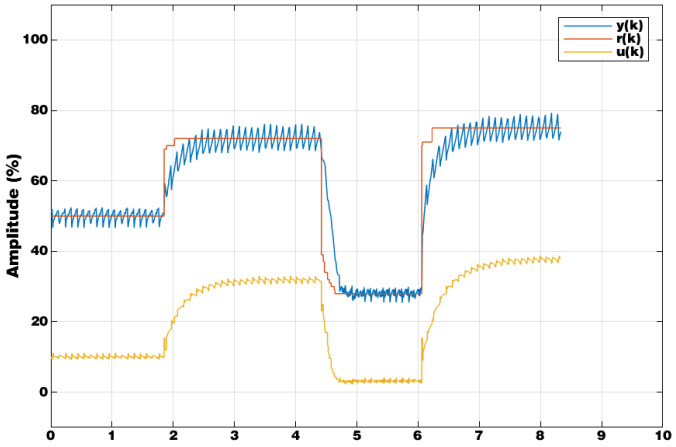

Fig. 10. Screenshot of the Simulink scope for PID control with interactive manual reference directly from the computer.

has been introduced here. As it was demonstrated by the sample experiments, the dynamic character of the closed-loop system allows students to learn about simple linear system modeling but also enables the curriculum to incorporate experiments aimed at exploring its nonlinear and hysteretic properties. The range of controller algorithms that can be tested using this device is practically endless and limited only by the underlying MCU hardware.

There are several aspects of the proposed hardware that may be improved in the future. For example, the optical tunnel may be equipped by another—preferably less powerful—LED that acts as a repeatable source of disturbance. A slightly more expensive but further refined variant of the same concept may replace the LDR to a calibrated device returning physical units of brightness in lux and also power the LEDs by an external DAC chip to reduce steady state oscillations caused by the PWM input.

The software accompanying the device may be augmented by functions implementing standard control algorithms or even system identification and modeling.

References

[1] A. Garrigós, D. Marroquí, J. M. Blanes, R. Gutiérrez, I. Blanquer, and M. Cantó, "Designing Arduino electronic shields: Experiences from secondary and university courses," in *2017 IEEE Global Engineering Education Conference (EDUCON)*, April 2017, pp. 934–937.

[2] H. M. Omar, "Enhancing automatic control learning through Arduino-based projects," *European Journal of Engineering Education*, vol. 43, no. 5, pp. 652–663, 2018.

[3] W. J. Esposito, F. A. Mujica, D. G. Garcia, and G. T. A. Kovacs, "The lab-in-a-box project: An Arduino compatible signals and electronics teaching system," in *2015 IEEE Signal Processing and Signal Processing Education Workshop (SP/SPE)*, Aug 2015, pp. 301–306.

[4] J. J. Castaneda, A. F. Ruiz-Olaya, W. Acuna, and A. Molano, "A low-cost Matlab-based educational platform for teaching robotics," in *2016 IEEE Colombian Conference on Robotics and Automation (CCRA)*, Sept 2016, pp. 1–6.

[5] I. Uyanik and B. Catalbas, "A low-cost feedback control systems laboratory setup via Arduino–Simulink interface," *Computer Applications in Engineering Education*, vol. 26, no. 3, pp. 718–726, 5 2018.

[6] P. Bakaráč, M. Kalúz, and Ľ. Čirka, "Design and development of a low-cost inverted pendulum for control education," in *Proceedings of the 21st International Conference on Process Control*, M. Fikar and M. Kvasnica, Eds., Slovak University of Technology in Bratislava. Štrbské Pleso, Slovakia: Slovak Chemical Library, June 6-9, 2017 2017, pp. 398–403.

[7] C. J. Bay and B. P. Rasmussen, "Exploring controls education: A re-configurable ball and plate platform kit," in *2016 American Control Conference (ACC)*, July 2016, pp. 6652–6657.

[8] S. C. McLoone and J. Maloco, "A cost-effective hardware-based laboratory solution for demonstrating PID control," in *2016 UKACC 11th International Conference on Control (CONTROL)*, Aug 2016, pp. 1–6.

[9] M. Kalúz, Ľ. Čirka, and M. Fikar, "Flexy: An open-source device for control education," in *13th APCA International Conference on Automatic Control and Soft Computing*, A. Cardoso, Ed., APCA. Univesrity of the Azores, Ponta Delgada, Portugal: Nova Gráfica, June 4-6 2018, pp. 37–42.

[10] M. Kalúz, "What is Flexy?" Online, 2017, [cited 26.09.2018]; GitHub Wiki page. Available from https://github.com/martin-kaluz/flexy-arduino.

[11] G. Takács, T. Konkoly, and M. Gulan, "Optoshield," Online, 2018, [cited 26.09.2018]; GitHub Wiki page. Available from https://github.com/gergelytakacs/AutomationShield/wiki/OptoShield.

[12] Sunrom Technologies, "Light Dependent Resistor - LDR," [online], 2008, datasheet for Model no. 3190; [cited 27.03.2019]; Available from https://www.sunrom.com/get/443700.

[13] RS Components, "Light dependent resistors," [online], 1997, datasheet 232-3816, data pack F, for NORP12 RS stock number 651-507 [cited 27.03.2019]; Available from http://www.bilimteknik.tubitak.gov.tr/sites/default/files/gelisim/elektronik/dosyalar/40/LDR_NSL19_M51.pdf.