Received: 16 December 2018 | Accepted: 19 February 2020

DOI: 10.1002/cae.22219

RESEARCH ARTICLE

WILEY

Fuzzy control of self-balancing robots: A control laboratory project

Ákos Odry[1] | Róbert Fullér[2] | Imre J. Rudas[3] | Péter Odry[1]

[1]Department of Control Engineering and Information Technology, University of Dunaújváros, Dunaújváros, Hungary

[2]Department of Informatics, Széchenyi István University, Győr, Hungary

[3]University Research, Innovation and Service Center, Óbuda University, Budapest, Hungary

Correspondence
Ákos Odry, University of Dunaújváros, Táncsics Mihály u. 1, 2400 Dunaújváros, Hungary.
Email: odrya@uniduna.hu

Funding information
EFOP, Grant/Award Numbers: 3.6.1-16-2016-00003, 3.5.1.-16-2017-00006, 4.2.1-16-2017-00011, 3.4.4.-16-0022, 3.6.1-16-2016-00010; GINOP, Grant/Award Number: 2.3.4-15-2016-00003; NTP-FKT, Grant/Award Number: 17-0011

Abstract

This paper presents a novel control laboratory project that provides hands-on experience in feedback control concepts (embedded control systems) through dedicated assignments, with a particular focus on the design and implementation of fuzzy control. The project is structured around an inexpensive, portable self-balancing robot (SBR), whose embedded system is realized using commercially available breakout boards as the first assignment. For the stabilization of the plant, students are guided to execute the essential stages of control system design, from system modeling and parameter optimization, over basic or advanced control strategy design in the MATLAB/Simulink environment, to both implementation and validation of the closed loop on the real robot. To demonstrate and foster the application of fuzzy logic, the second part of the paper introduces a simple control strategy based on fuzzy logic controllers. Then, a lookup table-based implementation technique is described for the demonstration of manual interfacing and embedded coding of fuzzy control strategies. The proposed methods are clear and straightforward; they highly foster the understanding of feedback control techniques and allow students to gain vast knowledge in the practical implementations of control systems.

KEYWORDS

Arduino, embedded control system, experiment-based teaching, fuzzy control, self-balancing robot

1 | INTRODUCTION

Both in theory and practice, transferring competitive knowledge is one of the most exciting challenges in engineering education. Laboratory experiments of interesting real-world problems effectively complement theoretical lectures, significantly improve the learning experience, and aid engineering students in recognizing their professional interests as well as enhance their creativity and team management skills. Moreover, the in-depth knowledge acquired during in situ experimentation highly facilitates the obtainment of positions in industry and motivates students to continue their research work at doctoral schools. These recognitions have led universities to continuously develop and include hands-on laboratories in their curricula [11,14,18,54,57]. An interesting and pragmatic project, carried out by groups of students in laboratory sessions, can bridge the gap between theory and practice, thereby providing a better understanding of fundamental concepts.

Control systems play an important role both in academics and industry. Therefore, control system design (CSD) is an essential course taught extensively in the field of electrical, mechatronic, and mechanical engineering at many universities. This multidisciplinary course is offered

at the master's level to students who have background knowledge of signals and systems, basics of automatic control, and, optionally, embedded systems. The main goals of the course are the following [54]:

1. Providing a methodological basis for the analysis of the controlled plants;
2. detailing some of the classical and advanced control techniques that may be used to control them;
3. enabling the students to implement the acquired knowledge in the control of real plants.

The lectures are usually complemented by practice sessions and laboratory experiments. The practice sessions aim to reinforce the theoretical concepts in a MATLAB/Simulink environment, the actual backbone of feedback control education. Furthermore, as all engineers need to have the essential skills to design and implement a digital control system (including knowledge of sensors, actuators, and real-time programming) [11], the goal of the laboratory experiments is to implement the techniques introduced in the lectures to design control structures for real plants. The laboratory sessions can be formed around a project work that is carried out by a small group of students week by week, thereby strengthening both social and professional skills in teamwork, problem-based thinking, and so forth.

Feedback control educators face the challenge of providing interesting hands-on experiments and control design tasks that increase students' interests and appreciation for feedback control. The following subsections discuss the characteristics, advantages, and disadvantages of different laboratory types.

1.1 | Traditional control laboratories

Classical CSD laboratories include textbook-type problems such as the ball-and-beam system [17]. To introduce students to this subject matter, new experiments and different face-to-face and remote laboratories have been described in the literature. The list of proposed experiments includes a magnetic levitation device and its neural network-based feedforward control [61], proportional integral derivative (PID) and linear quadratic regulator (LQR)-controlled inverted pendulums and coupled water tanks [13,62,65], and sliding mode-controlled (SMC) refrigeration systems [56]. An interesting real-world design problem of slip control of antilock braking systems (ABS) was proposed in [54], and the development of PI and fuzzy gain-scheduling controllers were included in the syllabus.

Robotics is an integral part of engineering education [20,27,42,66]. Therefore, mobile robots and unmanned aerial vehicles (UAV) have also turned into interesting pedagogical tools in CSD laboratories. In [60], a National Instruments Robotics Starter Kit robot was proposed to be used in teaching advanced control concepts such as the model predictive control (MPC) strategy. References [5] and [72] present PID-controlled quadrotor platforms for educational purposes.

Most of the aforementioned laboratory plants are commercially available as complete, ready-to-use laboratory kits from manufacturers such as Quanser, dSpace, and National Instruments, enabling the development of control algorithms easily using LabVIEW or MATLAB/Simulink. The advantages of these laboratories are as follows:

1) Students become acquainted with the design work flow used in many industries (real-time prototyping);
2) student work can be focused solely on the design, modification, and validation of control strategies through real-time simulation; and
3) implementation in just a click, that is, the automatic real-time C/C++ code generation allows running the model on the real hardware in seconds.

However, the three important disadvantages related to the traditional control laboratories are formulated below:

1) The laboratory kits require a dedicated space (due to their lack of portability); moreover, the adherent equipment, such as the real-time target machine and interface boxes, is rather expensive, which also prevents motivated students from constructing and developing their own embedded control system.
2) The development environment hides the embedded system architecture (including both the hardware and software parts as well as the implementation procedure) behind the analog and digital driver blocks of the employed input–output (IO) card. This prevents students from gaining experience in manual interfacing, embedded coding, and debugging of control algorithms (i.e., the implementation of embedded control systems remains unclear).
3) The laboratory experiments are dedicated, that is, modification of the experiment or software is impossible or heavily limited.

1.2 | Low-cost control laboratories

The result of the high cost and the need for dedicated hardware equipment is that only few plants are available in the aforementioned control laboratories. Moreover,

only a limited number of experiments can be performed, especially if the number of enrolled students is large. To solve these issues, the literature describes three possible solutions, namely simulation environments and software tools, affordable educational robotic platforms (e.g., LEGO Mindstorms), and development of digital signal processor (DSP) and microcontroller unit (MCU)-based inexpensive experimental apparatuses (e.g., Arduino-based control systems).

1.2.1 | Software tools

Interactive simulation environments, software tools, and virtual laboratories help students in improving their theoretical knowledge and in experiencing what they might face in the real world of engineering. The literature proposes state-space technique-based power system stabilizer design projects in MATLAB [10], graphical user interfaces as educational tools for teaching the basics of (neuro) fuzzy control of direct current (DC) motors [1,16] and induction machine drives [24], educational MATLAB/Simulink simulation platforms for both PID control of servo systems [3], and LQR and fuzzy logic-based adaptive PID control of quadcopters [35,58]. These software tools are valuable complementary materials, but simulation environments cannot replace experimentation with real plants [11,54,65].

1.2.2 | Robot kits

The shortcomings of software tools can be avoided with the application of affordable educational robotics kits in CSD laboratories. Reference [28] proposed a fuzzy control laboratory in which a PIC MCU-based Robo-PICA educational robot was programmed for line following. In [6] and [29], LEGO Mindstorms robots were used in control systems and robotics laboratories. In a similar vein, reference [4] proposed a LEGO Mindstorms-based self-stabilized bicycle for CSD laboratories and also presented LQR and Kalman filtering techniques. For LEGO Mindstorms, MATLAB/Simulink offers native support; therefore, controllers can be implemented through special LEGO library blocks. This approach allows both fast prototyping and online monitoring in Simulink. However, the practical implementation of control algorithms still remains elusive.

1.2.3 | Experimental apparatuses

Self-developed apparatuses are also popular in CSD laboratories. Reference [40] proposed a self-balancing human transportation vehicle (HTV) for teaching control systems. The HTV was built employing a DSP board and low-cost commercial components, and it was used to demonstrate the development and manual implementation of PID and phase lead–lag compensation techniques. A mechatronic aero pendulum and its phase lead–lag compensation in MATLAB/Simulink were also presented for CSD laboratories [15]. In [22], a dsPIC board-based DC motor control setup was proposed; moreover, the development and manual implementation of PID and two-degree-of-freedom control techniques were included in the syllabus. In addition, dsPIC board-based mobile robots were applied to increase student motivation toward automation and robotics, as given in [25].

Recently, low-cost open-source Arduino boards have prompted many researchers to develop CSD lab kits [7,20,30,52]. Arduino offers an easy-to-use integrated development environment (IDE), in which simple functions and libraries are available to use the MCU peripherals. Thus, students do not waste their time learning how to set up the registers; instead, they are able to focus on the implementation of the control algorithms. Furthermore, the flexibility and modularity allow both fast prototyping and wide interaction with sensors and actuators on breakout boards from third-party developers. Reference [63] presented Arduino-based mobile platforms for teaching basic control concepts such as system identification, PID design, and state estimation. Moreover, the Arduino Engineering Kit provides a complete embedded environment, in which students can learn about the key aspects of mechatronics, MCU peripherals, and MATLAB and Simulink programming via the realization of three different Arduino-based projects.

1.3 | Self-balancing robot as a lab kit

As the descendant of the pendulum-cart systems, the self-balancing robot (SBR) provides a wide application spectrum due to its advantageous electromechanical properties, such as the compact construction, small footprint, zero turning radius, low cost, and low energy consumption [8,39]. In the field of education and research, the SBR is considered as a mobile platform to be controlled effectively as much as an important benchmark to verify the control approaches. These advantages, along with the successful commercial product Segway PT [59], can make a good impression on students [19]. Therefore, the SBR has a high potential as a pedagogical tool to reinforce feedback control concepts.

Numerous control approaches have been proposed for SBR systems, from simple PID [37] and state feedback techniques [21,31,36], over the advanced SMC [12,23,70]

and $\mathcal{H}_\infty$ [55] control methods, to adaptive [64] and soft-computing techniques [26,69]. Most of these methods contain complex formulations that make the application and implementation seem exceedingly complicated for students. Therefore, a fuzzy control approach for SBR systems, which is less complex, easy to implement in embedded environments, and can be commonly used both in education and practice, still remains an important issue to be further addressed.

1.4 | Contribution of the paper

The first part of the paper proposes a novel, low-cost CSD (and robotics) lab project in which project-based learning is implemented. A simple, breadboard-based SBR frame, an Arduino board and different commercially available breakout boards are given to the students, and the aim of the project task is to build and stabilize the plant. This project fosters students to apply many areas of knowledge, including embedded system design, modeling, identification, feedback control design and MCU programming. Despite the MATLAB/Simulink native support for Arduino, students are required to develop their handwritten, code-based implementation. The authors believe that the code, developed step-by-step, makes it possible to understand better the aspects of embedded control systems and also to dispel the illusion related to the practical implementation of control algorithms. Moreover, the applied project-based learning can motivate students toward automation and robotics [33,52]. Students accomplishing the project gain an in-depth CSD knowledge enriched by experience in electronics, software and mechanical systems. This competitive knowledge has proven to be an excellent starting point towards a wide range of industrial and academic careers [11,18].

The second part of the paper proposes a straightforward fuzzy control strategy (FCS) for SBR systems and a LUT-based implementation technique of fuzzy logic controllers (FLCs) for educational purposes. The authors have noticed in teaching CSD that students are reluctant to apply intelligent control methods such as fuzzy control. This outcome can be related to the hazy, "fuzzy" perception of students in terms of the practical implementation of FLCs. Namely, most students learn the design steps (using the Fuzzy Logic Toolbox of MATLAB), yet they remain hesitant about the implementation on real hardware. This realization motivated the authors to propose a LUT-based implementation technique of FLCs that can be used in education. The technique is easy to implement, requires a small amount of calculations and is suitable for small embedded processors such as Arduinos. The implementation technique along with the proposed control strategy will increase student interest and motivation toward fuzzy control.

The differences between similar CSD laboratories [4,28,40,54] and the novelties of this paper are summarized as follows:

1) The proposed lab kit is a small-sized, carry-home mobile platform. The breadboard-based frame enables the students to gain experience in MCU-based systems. The carry-home characteristics allow students to perform experiments wherever and whenever they may wish. Finally, the low-cost Arduino and breakout boards provide both an easy-to-use programming environment and easy-to-interface hardware capabilities.
2) The proposed FCS is featured specifically for education (i.e., simple structure and easy implementation). An expert-oriented design approach is introduced that uses those simple heuristic knowledge-oriented tools that fuzzy logic meant to offer.
3) A simple LUT-based implementation technique of FLCs is described. The native support of MATLAB/Simulink prevents students from obtaining experience in manual implementation. On the other hand, the direct implementation (fuzzification, inference, and defuzzification) of FLCs may appear as overly complicated for students because it has significant computational complexity. The proposed LUT-based technique is easy to apply and requires a small number of calculations, thereby bridging the gap between the simulation and embedded environment in a clear, comprehensible way.
4) The proposed CSD project cancels the disadvantages specified in the previous subsections and facilitates the acquisition of the complete knowledge described in [11,18], which is indispensable for future industrial control engineers as well as for students aiming to become academics.
5) Both the SBR lab kit and CSD project can be replicated in the laboratory of any institution.

An extensive survey of existing labs revealed no match with the proposed setup and assignments. The paper systematically describes the construction, development and fuzzy control of the proposed SBR for teaching CSD in laboratories. All the information, including the computer-aided design (CAD) models, MATLAB/Simulink files, and MCU software have been made publicly available in the supplementary online material [43] to help other lab teams in designing similar experiments.

The remainder of the paper is organized as follows. Section 2 introduces the SBR platform, while Section 3 describes the derivation of its mathematical model. In Sections 4 and 5, the authors discuss the FLC design steps

and the implementation procedure, respectively. In Section 6, the assessment and curriculum plans are described. Finally, Section 7 provides conclusions and recommendations for future developments.

2 | SELF-BALANCING ROBOT PLATFORM

The proposed SBR kit is considered as a simple yet versatile pedagogical tool that triggers students to use their embedded system skills and apply dynamic system analysis, feedback control, simulation evaluation, and experimentation.

2.1 | Project assignment

The first assignment has the students build the robot and initialize the MCU peripherals. Every group of students has an SBR frame, the necessary breakout boards and supplementary materials at their disposal. Moreover, students are given a prelaboratory assignment of studying the laboratory manual, which describes the hardware and software parts, the MCU peripherals and basic embedded code templates.

2.2 | Mechatronic construction

The mechanical structure of the SBR consists of two actuated wheels and a 3D printed inner body (IB) that forms the inverted pendulum. The IB is equipped with two breadboards that support the realization of the embedded electronics. The wheels are driven with DC motors that are attached to the IB. Figure 1 displays the SBR both in its assembled (a) and disassembled (b) states. The dimensions (length, height, and width) of the chassis are 165, 99, and 171 mm, respectively, complete with 64-mm diameter wheels.

The embedded electronics is built around an Arduino Nano, which is a low-cost, small, and breadboard-friendly development board based on the ATmega328 MCU. The InvenSense MPU-6050 contains accelerometer and gyroscope sensors, and moreover, is available on commercially available breakout boards and is applied to measure the IB dynamics. This inertial measurement unit (IMU) is interfaced with the Arduino via the I2C-bus and provides 3D acceleration and angular rate measurements. The actuators are 12 V DC motors (model No. GB37Y3530-12V-251R) equipped with both planetary gearheads of 43:8 reduction ratio and two-channel incremental encoders of 16 lines per revolution resolution. The encoder

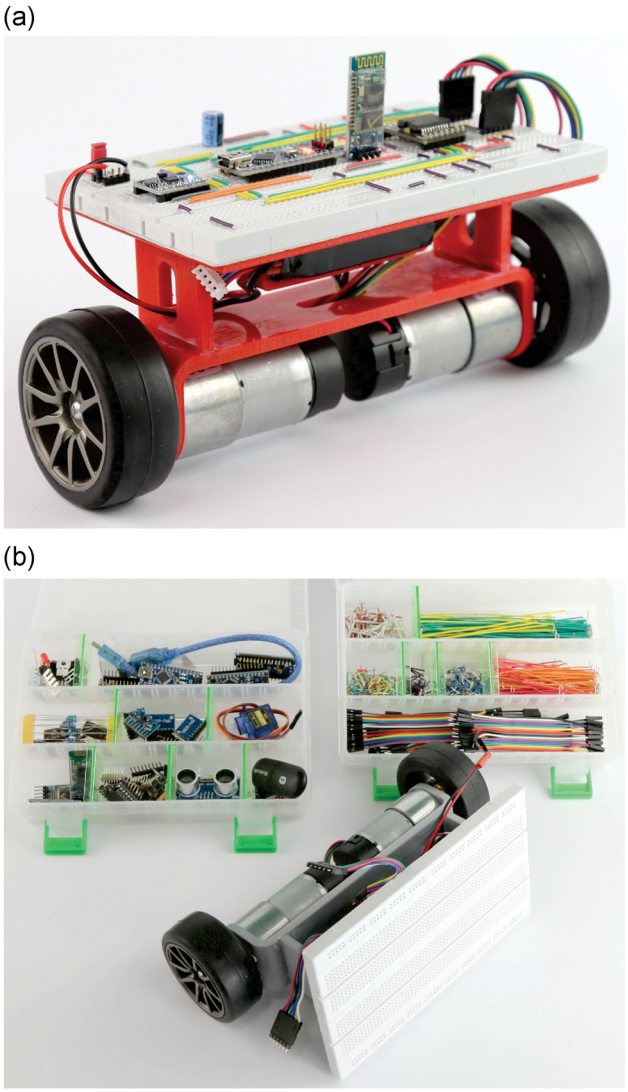


FIGURE 1 Photograph of the SBR platform: (a) assembled version, (b) disassembled version. SBR, self-balancing robot

signals are processed by two digital inputs, one of which triggers interrupts on rising or/and falling edges. The motors are driven with pulse-width modulation (PWM) signals via an L298P dual H-bridge breakout board. For advanced students, an HC-05 Bluetooth module is provided for wireless data transfer and is attached to the universal asynchronous receiver-transmitter (UART) peripheral of the MCU. The electronic system is supplied by a three cell lithium battery pack. Figure 2 depicts the circuit diagram (a) and its possible realization on breadboards (b). The carefully selected hardware components amount to an overall cost of less than 180 US$ for the setup.

Once the embedded electronics are realized on the breadboards, students are required to initialize the MCU peripherals and create a minimal working environment in the Arduino IDE. This environment drives the motors

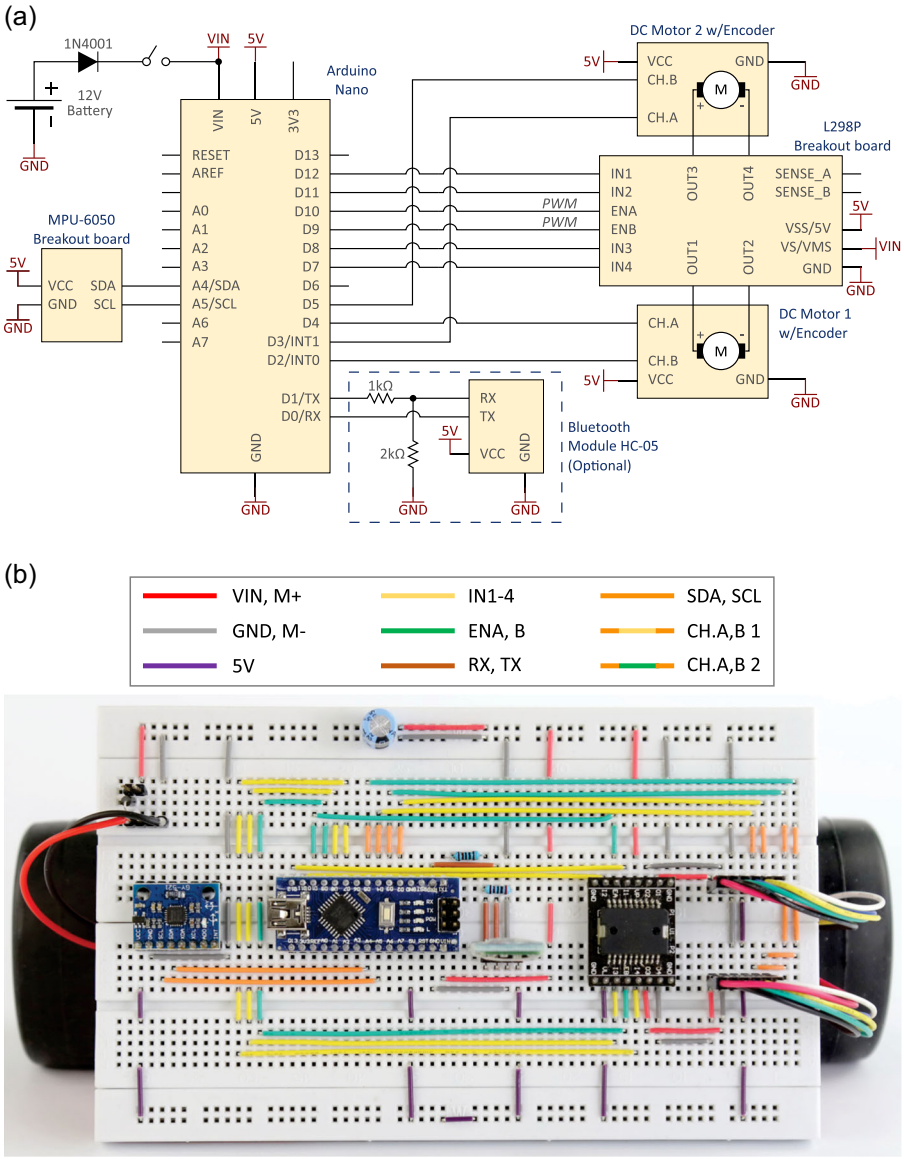


FIGURE 2 Embedded electronics of the SBR: (a) circuit diagram and (b) a possible realization. DC, direct current; SBR, self-balancing robot

in both directions, and it reads and sends the instantaneous IMU (acceleration and angular rate) and encoder (motor shaft position) measurements to the PC. The authors suggest the employment of X1 encoding for the determination of the shaft positions.

2.3 | Project objectives

At this stage of the project, the main objectives are (a) to help students brush up on their knowledge of the materials related to MCU programming and (b) to familiarize them with the basics of sensing and actuating. Students are expected to gain experience in using the basic MCU peripherals from digital I/O, over the PWM technique and interrupt handling, to serial communication such as UART and I2C. Advanced students can extend the embedded electronics in Figure 2 with shunt resistors (via the SENSE pins of L298P) to measure the motor currents by means of the analog-to-digital converter (ADC) of the Arduino.

3 | MATHEMATICAL MODEL

To be able to efficiently design the control algorithms, the mathematical model of the SBR has to be obtained first [53]. A straightforward 8-dimensional state-space model is proposed, which includes the motor dynamics and the terminal voltages from the input (or control) signals.

3.1 | Project assignment

At this point, the students' tasks include (a) to both derive and implement the mathematical model of the SBR in

MATLAB/Simulink and (b) to estimate the numerical values of the missing plant parameters. Students are encouraged to split the problem into two subproblems, namely, the modeling and parameter estimation of the DC motor should be solved first, the SBR model is then derived, and its unknown parameters are obtained. The laboratory manual describes most of the SBR parameters with technical drawings; moreover, both figures and model equations facilitate the derivation of the system dynamics. Table 1 is also provided to students and contains both the directly measurable parameters (e.g., wheel radius) and the presumable value of the viscous friction coefficient between the wheels and the contact surface. For the estimation of the unknown motor parameters, the Simulink Design Optimization tool is suggested for the students. Regarding the unknown IB parameters, two solutions are recommended; namely, the parameters are either calculated based on a simple CAD model, or the SBR body is decomposed into simple body elements, and the superposition of moments is performed.

3.2 | DC motor model

The gear motor system is characterized by the parameters given in Table 2. The gear ratio (k) defines the connection between the rotor (motor) and gear (load) sides, that is, the angular position of the rotor is $\theta_r = k\theta_g$, where θ_g denotes the shaft position on the gear side. The fundamental electrical and mechanical equations on the load side are summarized as [46,49]

$$u = \mathrm{RI} + \mathrm{L}\dot{I} + k_\mathrm{E} k\dot{\theta}_\mathrm{g},\ J_\mathrm{M}\ddot{\theta}_\mathrm{g} = k\tau_\mathrm{M} - f_\mathrm{M}\dot{\theta}_\mathrm{g} - \tau_\mathrm{ext}, \quad (1)$$

where τ_M denotes the torque transmitted to the shaft and is proportional with the rotor current (I), that is, $\tau_\mathrm{M} = k_\mathrm{M} I$. Moreover, u and τ_ext denote the terminal voltage and external torque, respectively. The terminal voltage of the motor is proportional with the PWM value (d_PWM, $u \propto d_\mathrm{PWM}$), that drives the H-bridge.

TABLE 1 Notation of basic robot parameters

Parameter	Symbol (unit)	Value
Mass of the wheels	m_w (g)	33.8
Radius of the wheels	r (mm)	32
Distance between the wheels	d (mm)	205
Total mass of the IB	m_b (g)	802.2
Viscous friction coefficient (wheel-surface)	f_w (mNm/s)	0.54
Gearbox ratio	k	43.8

TABLE 2 Estimated values of the direct current motor parameters

Parameter	Symbol (unit)	Initial	Estimated
Rotor resistance	$R(\Omega)$	5	4.01
Rotor inductance	$L(\mu\mathrm{H})$	10	22.22
Total motor inertia	$J_\mathrm{M}(\mathrm{kgmm}^2)$	0.8	2.74
Total viscous friction	$f_\mathrm{M}(\mu\mathrm{Nm/s})$	20	7.27
Torque constant	$k_\mathrm{M}(\mathrm{mNm/A})$	10	5.44
Back-EMF constant	$k_\mathrm{E}(\mathrm{mVs/rad})$	10	5.44

Students are offered different approaches to implement the model in MATLAB/Simulink. Either a state-space representation is determined and the state-space block is employed, or simple integrator, gain and sum blocks are used to implement Equation (1). Additionally, Laplace transform can also be performed to determine the transfer functions, or the Simscape modeling blocks are offered alternatively. A feasible implementation of Equation (1) is illustrated in Figure 3.

The unknown motor parameters ought to be obtained by means of the Parameter Estimation tool of Simulink. The simplest but most practicable estimation approach is based on the measured step response data of the DC motor. Namely, the Arduino is programmed to both perform step excitation and continuously send the measured motor speeds via UART, while MATLAB/Simulink is prepared to collect the measurements and employ them in Simulink Design Optimization. Students are expected to experiment with different cost functions and optimization methods. This procedure serves as a demonstration for students that (a) the cost function quantifies the estimation quality, (b) the parameters are optimized through finding the extrema of the cost function, and (c) optimization algorithms effectively speed up the tuning of model parameters based on the measured data. Advanced students are encouraged to extend Equation (1) with additional friction models and nonlinearities, such as dead-zone or stick-slip effects, and estimate the parameters of a more realistic motor model by performing more specified experiments.

Table 2 summarizes the parameter estimation results, while Figure 4 depicts the comparison of DC motor simulation output with the measured data set. The estimation tool executed the nonlinear least-squares optimization, which minimized the sum squared error (cost function) between the measurement and simulation results. In the first row of Figure 4, the simulation results are shown both before (red curve) and after (yellow curve) the optimization, and the simulation results are compared to the measured data

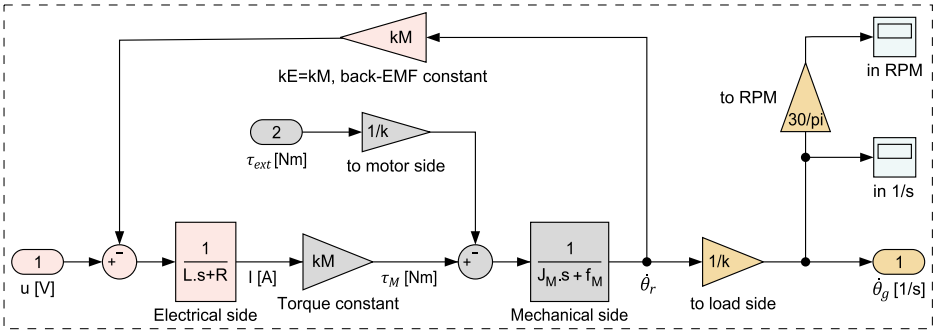


FIGURE 3 A possible direct current motor model implementation in MATLAB/Simulink

(blue curve). In the second row, the applied step excitation is displayed. The measurements were collected at $f_s = 50$ Hz sampling frequency. It can be concluded that the curve for the optimized motor parameters fits with the measured data to a satisfactory degree, and thereby a simple but still realistic DC motor model is determined by the proposed estimation procedure. At this point, students move on to the determination of the IB dynamics.

3.3 | Robot dynamics

A comprehensive picture of the robot control system is provided in Figure 5, where the structure of both motor and IB models can be identified in detail.

The dynamics of the IB coupled with the motor gear model (1) together form the dynamic model of the whole SBR system. The derivation of system dynamics starts with the determination of the total kinetic (translational and rotational) and potential energies of both the wheels and the IB. Based on the kinetic (K) and potential (P) energies, the Lagrangian of the system is formed as $\mathcal{L}(q) = K - P$ and given in the appendix [46,49]. Let $\theta_{1,2}$ and θ_3 indicate the angular displacement of the wheels and the angular position of the IB (i.e., pendulum angle), respectively. The configuration variables of the system are taken as $q = (\theta_1, \theta_2, \theta_3)^T$. Using the Euler–Lagrange formulation, the equations of motion of the SBR system are determined as

$$M(q)\ddot{q} + V(q, \dot{q}) = \tau_a - \tau_f, \tag{2}$$

where τ_a denotes the transmitted torques by the DC motors, and τ_f indicates the effect of friction; furthermore, $M(q)$ is the 3×3 inertia matrix, and $V(q, \dot{q})$ is the

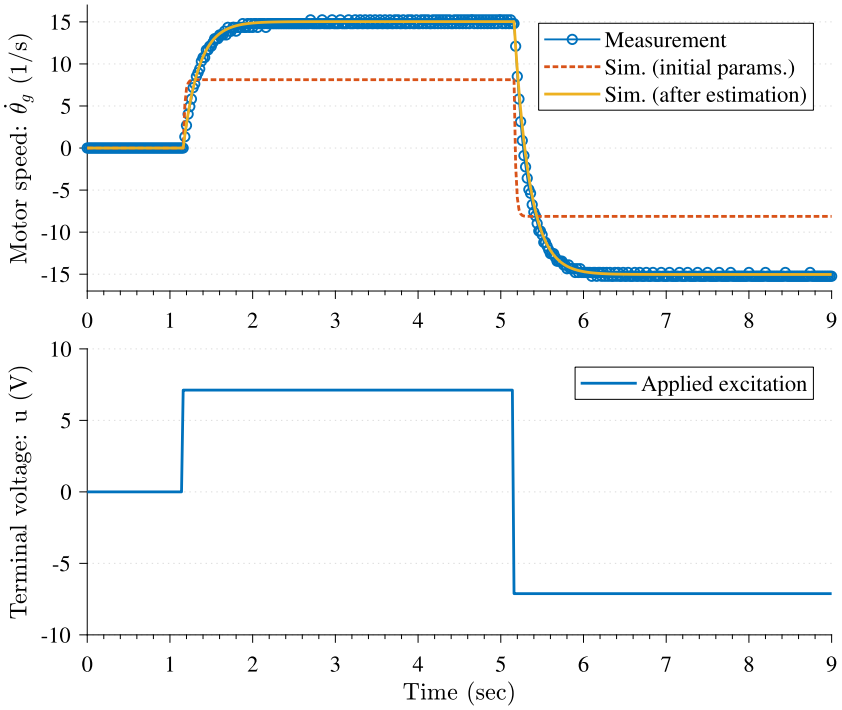


FIGURE 4 Direct current motor simulation with initial and estimated parameters: motor speeds and applied excitation

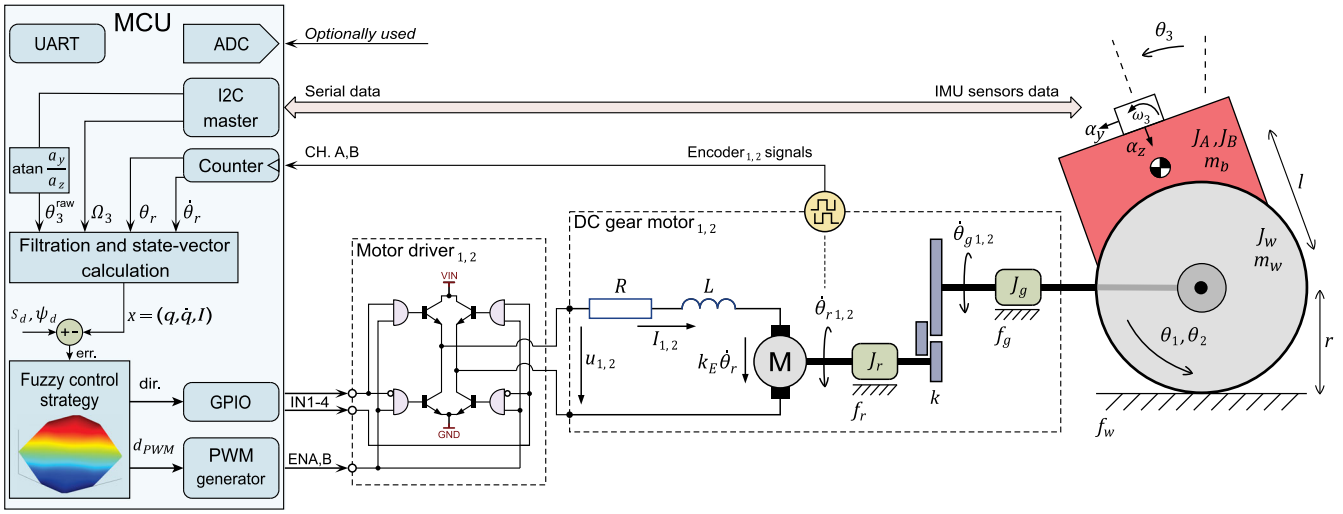


FIGURE 5 Block diagram of the robot control system. UART, universal asynchronous receiver-transmitter

3-dimensional vector term including the Coriolis, centrifugal, and potential force terms. The parameters of these matrices are described in [46,49].

Students are expected to understand the coupling between Equations (1) and (2). Two gear motors are attached to the IB; therefore, based on Equation (1), two gear motor models are required to be defined with $u := (u_1, u_2)^T$, $I := (I_1, I_2)^T$, $\theta_g := (\theta_{g,1}, \theta_{g,2})^T$, and $\tau_M := (\tau_{M,1}, \tau_{M,2})^T$, where the subscript number refers to the first or second DC motor. Moreover, it has to be recognized that the relationship between the angular positions of the wheels (θ_1, θ_2) and the pendulum angle (θ_3) is given by the motor shaft position (θ_g), namely, $\theta_{g,i} = \theta_i - \theta_3$, $i = \{1,2\}$. Taking these considerations into account, the torque difference in Equation (2) is described in the Appendix. Rearranging Equations (1) and (2), the state-space representation $\dot{x}(t) = h(x, u)$ of the SBR is obtained, where the state vector is defined as $x = (q, \dot{q}, I)^T$ [46,49]:

$$\dot{x}(t) = \begin{bmatrix} \dot{q} \\ M(q)^{-1}(\tau_a - \tau_f - V(q, \dot{q})) \\ \frac{1}{L}\left(u - k_E k \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \end{bmatrix} \dot{q} - \mathrm{RI}\right) \\ y(t) = \mathrm{C}x(t), \end{bmatrix}, \quad (3)$$

where C is selected to produce the output vector as $y = (s, v, \theta_3, \omega_3, \psi, \xi, I_1, I_2)^T$, the linear displacement and speed of the plant are indicated with $s = r(\theta_1 + \theta_2)/2$ and $v = \dot{s}$, respectively, while $\omega_3 = \dot{\theta}_3$ is the angular velocity of the IB, and the yaw angle and rate are defined as $\psi = r(\theta_2 - \theta_1)/d$ and $\xi = \dot{\psi}$, respectively.

At this point, students are offered to either linearize Equation (3) and implement the system dynamics with the Simulink state-space block or implement Equation (3) directly by means of a system-function (S-function) block. Then, the unknown robot parameters have to be obtained to simulate the SBR dynamic behavior; these are the center of mass location (l) and the inertia values J_A and J_B (the remainder are given in Table 1). Mechanical and mechatronic engineering students are encouraged to estimate the unknown parameters by means of CAD software, such as Solidworks. Electrical engineering students are given a chance to gain experience in CAD, but they are also encouraged to decompose the SBR into body elements and apply the superposition of moments to estimate l, J_A, and J_B. Figure 6 shows a simple CAD model of the SBR created by a mechatronic engineering student, while the inertia parameters estimated with both the CAD software and multibody approximation are summarized in Table 3. The

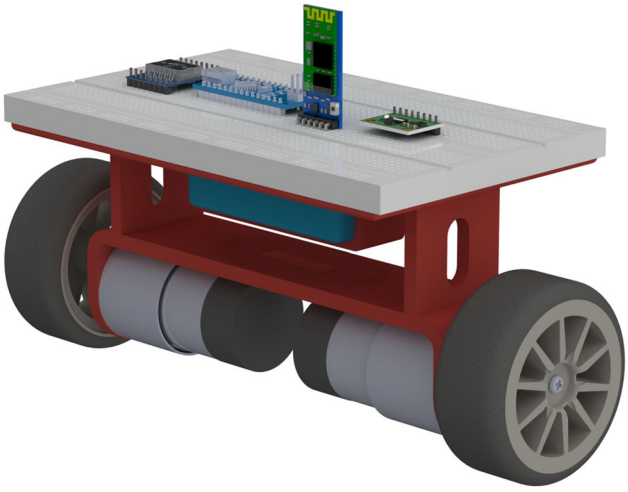


FIGURE 6 Simple computer-aided design model of the SBR platform (student work)

Parameter	Symbol (unit)	CAD	Multibody
Center of mass location from wheel axle (A)	l(mm)	19.77	23.33
Moment of inertia of the IB about the wheel axle (A)	J_A(kgmm^2)	1309.19	1300.00
Moment of inertia of the IB about its vertical axis (B)	J_B(kgmm^2)	2855.89	2149.64

Abbreviation: CAD, computer-aided design.

TABLE 3 Estimated inner body parameters: Computer-aided design and superposition-based results

dynamic model (3) implemented in MATLAB/Simulink facilitates the design and testing of the control algorithms before the concrete realization on the real SBR.

3.4 | Project objectives

Both mathematical modeling and parameter estimation are important stages in control system development. The assignment objectives are the following: (a) to illustrate to students the main engineering considerations and formulations employed in modeling real dynamical systems; (b) to both demonstrate different parameter estimation approaches and allow students to set up the appropriate experiments for model validation; (c) to make students familiar with the modeling and simulation capabilities of MATLAB/Simulink; and (d) to foster understanding of both the DC motor and SBR based on the computer-aided simulations.

4 | FUZZY CONTROL STRATEGY

Zadeh's fuzzy logic has introduced a linguistic information-based CSD perspective that allows the system designer to develop the feedback loop employing simple IF-THEN linguistic rules [68]. Moreover, the inference mechanism formed by these fuzzy rules facilitates effective development even if an inaccurate mathematical model or black-box object is available [71]. The control scheme proposed in this section is elaborated with respect to the aforementioned advantages.

4.1 | Project assignment

The students are tasked with developing a FCS that simultaneously ensures the planar (longitudinal and rotational) motion of the SBR and stabilizes its IB around the unstable upright position. Students are required to (a) make deductions related to the system dynamics, (b) both define linguistic variables and form heuristic IF-THEN rules, and (c) create FLCs with the help of the MATLAB Fuzzy Logic Toolbox. Splitting the control problem into separate tasks is recommended such that the back and forth motion, yaw angle control and IB stabilization problems are elaborated separately. Students are expected to both experiment with different membership functions, inference properties and defuzzification methods and carry out simulations in MATLAB/Simulink that illustrate successful feedback control. The laboratory manual contains references related to fuzzy logic and the usage of the Fuzzy Logic Toolbox. Moreover, students are given the main advantages of fuzzy control compared to the fundamental linear solutions they learnt at the lectures of basics of automatic control.

4.2 | Control system design

Fuzzy sets replace the classical two-valued logic with membership values that cover the whole continuous interval [0,1]. The fuzzy sets employed in the inference mechanism of the FLC contribute both to handle vagueness inherently included in the system and to achieve smooth control action. The block diagram of the FLC is depicted in Figure 7, while its main parts are described in [68].

4.2.1 | Control task

Let s_d and ψ_d define the desired linear displacement (position) and yaw angle of the plant, respectively; then, the control tasks are (a) $\lim_{t\to\infty}\theta_3(t) = 0$ for the IB angle, (b) $\lim_{t\to\infty}s(t) = s_d$ for the linear displacement, and (c) $\lim_{t\to\infty}\psi(t) = \psi_d$ for the yaw angle. The control scheme is required to drive the DC motors in such a way that the aforementioned tasks are accomplished simultaneously.

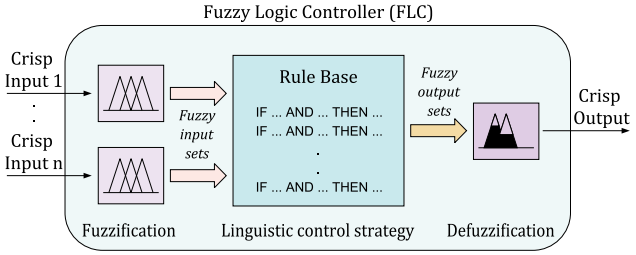


FIGURE 7 Structure of the fuzzy logic controller

4.2.2 | Control scheme

The control strategy elaboration consists of the construction of a control scheme and the definition of FLCs. As fuzzy logic enables the expert to design the control action based on IF-THEN rules, the students are asked to start the elaboration with an understanding of the system behavior. To stabilize the system, the following deductions have to be understood by the students.

1. IF the IB falls to one direction, THEN the wheels will be driven towards the same direction to bring the pendulum back into an upright position.
2. A similar condition is valid for the position stabilization of the wheels. Namely, IF the position tends to leave the desired value in one direction, THEN the wheels shall be driven in the opposite direction.
3. The yaw orientation is compensated as IF it differs from the desired value, THEN the two motors are driven in opposite directions.

Additionally, the deductions mentioned above can be expanded by taking into account both the IB and wheel velocities. By incorporating the velocities, more sophisticated rules are collected, and PD-type FLCs are formed. As each control objective can be accomplished with one dedicated PD-type FLC, the control scheme is constructed by combining three FLCs (hereinafter FLC1, FLC2, and FLC3). Figure 8 depicts a transparent and easily comprehensible fuzzy control scheme recommended for educational purposes. Students are given the corresponding Simulink model with empty FLC blocks, which are required to be replaced in the assignment.

The authors suggest the application of two input one output PD-type FLCs due to multiple reasons. On the one hand, the characterization of these FLCs is straightforward, and every stage of the algorithm (from fuzzification, over the rule base, and inference machine definition, to defuzzification) can be demonstrated. Moreover, students can easily utilize their background knowledge of PID control in the construction of the fuzzy inference mechanism. On the other hand, through the application of these simple PD-type FLCs, students are shown that the flexibility of fuzzy logic allows the definition of nonlinear control action effectively based on heuristic knowledge and without the application of complex mathematical formulations. The students can be further motivated by presenting a comparison of control actions related to the linear PD controller and nonlinear PD -type FLC. Complex FLCs (e.g., with more than two IOs) are not suggested for educational purposes because they cause students to struggle with the definition of both the IO relationships and fuzzy rule base; moreover, their tuning requires deeper expert knowledge.

4.2.3 | FLC properties

Based on Figure 8, the roles of each FLC and their IO signals are identified. Students are encouraged to understand the system dynamics, have experiences, and come to conclusions that can be used to heuristically define the linguistic variables and the universes of discourse of the IO variables. A possible selection of design properties and parameters for FLC1–3 are discussed as follows.

FLC1 is responsible for the robot position stabilization. The inputs of the controller (antecedents of each rule) are the position or tracking error $e_s(i) = s_d(i) - s(i)$ and its time derivative $e_v(i)$, while the output (consequent) is the command voltage $u_s(i)$ supplied to both motors. As the physical quantities are sampled with fixed $t_s = 1/f_s$ sampling time, as an example, $e_s(i)$ denotes the discrete time domain equivalent of the sampled signal $e_s(t)$:$e_s(i) = e_s(it_s)$. FLC2 ensures the balance control of the IB around the upright position. According to Figure 8, the pendulum angle error $e_{\theta_3}(i) = 0 - \theta_3(i)$ and its time derivative $e_{\omega_3}(i)$ are the inputs, while the control action $u_{\theta_3}(i)$ forms the output of FLC2. FLC3 corrects the yaw angle (orientation) of the robot. The inputs of the controller are the yaw angle error $e_\psi(i) = \psi_d(i) - \psi(i)$ and its time derivative $e_\xi(i)$, while the FLC output is the excitation voltage $u_\psi(i)$ that compensates the error.

The employed controllers are PD-type FLCs that produce a crisp output based on both the defined IO membership functions and the selected inference mechanism (implication, aggregation, and defuzzification methods). The FLCs are defined with Mamdani architecture IF-THEN rules. For the sake of simplicity, these FLCs are characterized by the same inference mechanism properties. The membership functions of the input variables were chosen with triangular shapes. Three membership functions were chosen for the inputs, being uniformly distributed across their universes of discourse (see Figure 9). The linguistic variables are N, Z, and P, which correspond to negative, zero, and positive fuzzy sets, respectively. For the outputs, singleton fuzzy sets were defined similarly with N, Z, and P variables. The IF-THEN rules of the PD-type FLCs and the selected inference mechanism are summarized in Table 4.

The proposed FLC architectures execute weighted average defuzzification for the crisp output calculation [47,50]. Based on Figure 8, the fuzzy control scheme produces the following excitation voltages to the motors:

$$u_i = u_s + u_{\theta_3} \pm u_\psi, i = \{1, 2\}, \tag{4}$$

where u_s, u_{θ_3}, and u_ψ denote the crisp outputs of the PD-type FLCs.

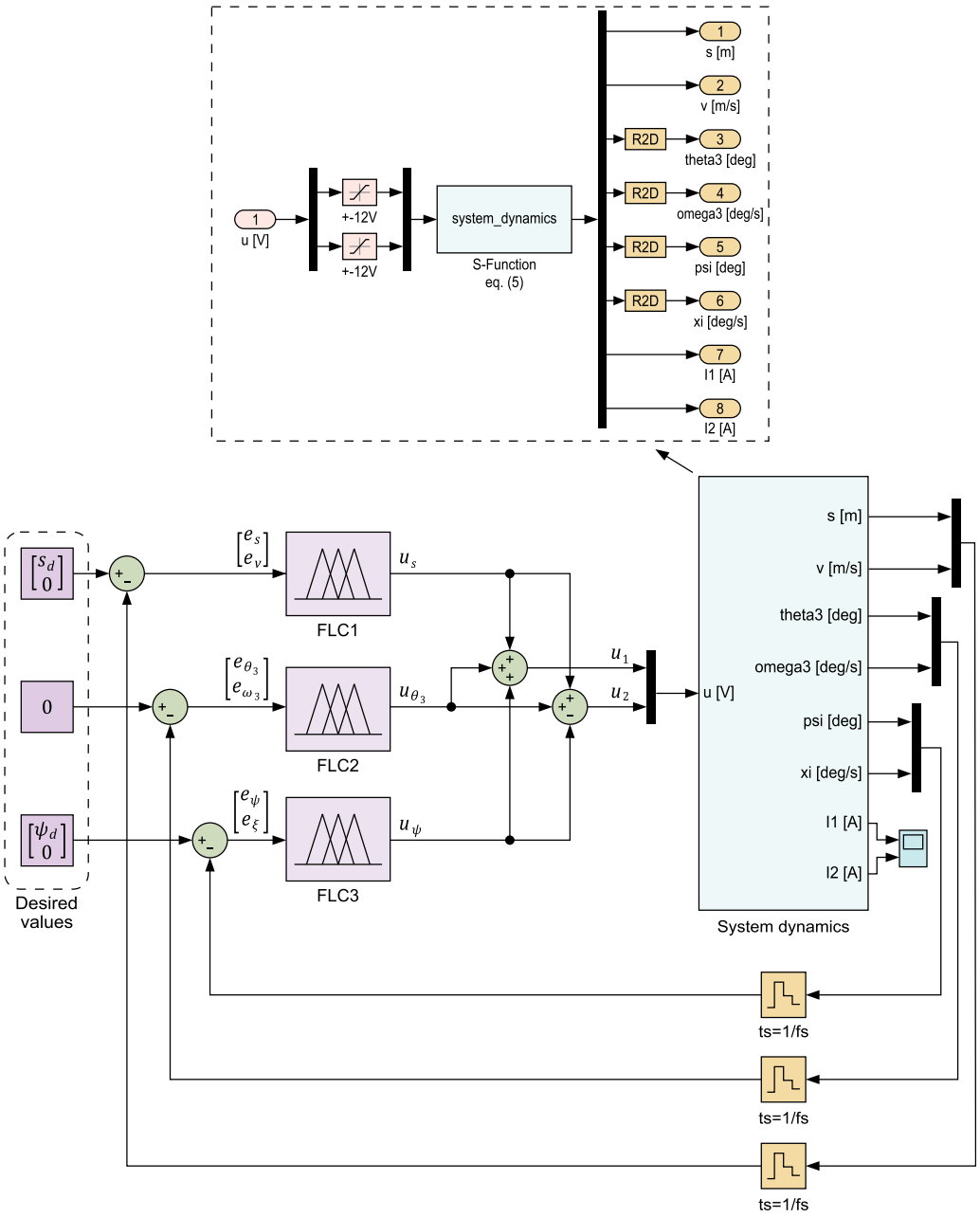


FIGURE 8 The elaborated fuzzy control scheme in MATLAB/Simulink

4.2.4 | Simulation results

The simulation of the control scheme is performed in MATLAB/Simulink. The state-space model (3) was implemented in an S-function block, while FLC1–3 were defined with the parameters described previously. The simulation model also takes into account the t_s sampling time.

Figure 10 depicts the simulation results, where the initial states of the state-space model (3) were defined as $x(0) = (-2.7, -4.7, 0.01, -2, 0.5, -0.7, 0, 0)^T$ (in SI units). From the top, the first row illustrates the SBR position $s(t)$ compared with the reference value s_d (position control performance), the second row indicates the yaw angle compensation ($\psi(t)$ and ψ_d), while the third row shows the inverted pendulum stabilization performance (pendulum angle $\theta_3(t)$). Additionally, the fourth row of Figure 10 displays the applied motor voltages during the stabilization. The desired values were provided via a signal generator block, namely, $s_d = (0, 170, 28, -115)$ mm robot positions and $\psi_d = (0, 60, -16)$ deg yaw angles were selected as desired values in the simulation.

From the simulation results, it can be concluded that the developed FCS successfully stabilizes the unstable

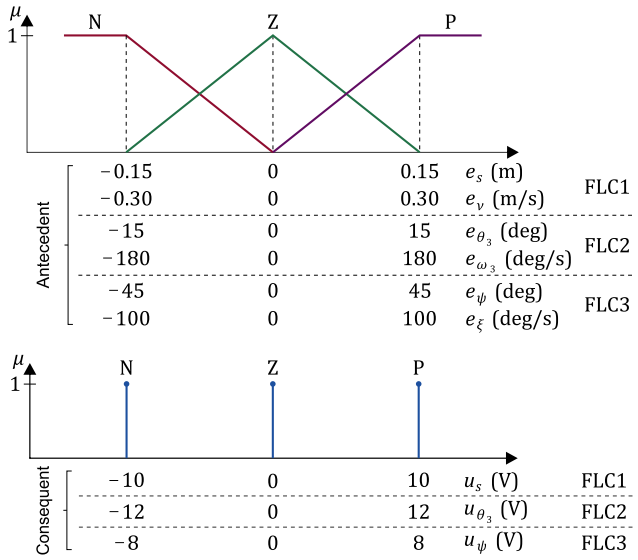


FIGURE 9 Membership functions of the IO variables of FLC1–3. FLC, fuzzy logic controller; IO, input–output

SBR system. The applied control scheme simultaneously ensures both the position and orientation of the plant and keeps the pendulum (IB) around the unstable upright position. The achieved control performance is not optimal, as the controllers were realized with heuristic knowledge-based intuitive design steps. At this point, students are suggested to tune the FLCs (i.e., to change the membership functions, the ranges of IO variables or the defuzzification method) and analyze the impact on the closed loop behavior, thereby enhancing the overall control performance experimentally.

4.3 | Project objectives

The project assignment aims to reinforce the concepts related to fuzzy logic and fuzzy control with the aid of the Fuzzy Logic Toolbox of MATLAB. The design project encourages the students to (a) acquire the heuristic knowledge-based design method fuzzy logic provides, (b) become acquainted with FLCs and the Fuzzy Logic Toolbox via solving the control problems, and (c) both analyze and tune control systems with the help of the MATLAB/Simulink simulation environment. This way, students are exposed to the advantages of fuzzy control, that is, that by employing fuzzy sets and approximate reasoning, successful and satisfying control is achieved easily and effectively. The computer-aided simulations help students in understanding what to expect in practice, and more importantly, the hands-on experience gained via constructing the control system also involves an initial notion of the firmware code to be written during the implementation process.

5 | IMPLEMENTATION

The implementation process includes writing the firmware code and embedded system testing procedures followed by the experimentation on SBR and the analysis of the real closed loop behavior. Efficient calculation is crucial in control systems with small embedded processors. Therefore, a LUT-based implementation technique of FLCs is proposed, which requires short computational time and presents a suitable solution for teaching the practical implementation, as well as the manual interfacing of FLCs.

5.1 | Project assignment

In the last assignment, students are required to both implement the elaborated control scheme in Arduino and demonstrate successful SBR control. Students are taught the direct and LUT-based FLC implementation methods and are then recommended to write an FLC function, which takes the error signals (derived from the measurements) as input parameters and returns the corresponding excitation. Next, the students are required to code the control scheme depicted in Figure 8 by determining the measurement data, utilizing the developed FLC function and employing Equation (4). Finally, the demonstration of successful stabilization is required; moreover, measurement results of the dynamic behavior have to be collected for documentation. Students are asked to analyze the differences between simulation and measurement data and to brainstorm about the methods

TABLE 4 Rule base and properties of the employed PD-type FLCs

Rule base					Properties	
Consequent $\{u_s, u_{\theta_3}, u_\psi\}$		Antecedent$_2$ $\{e_v, e_{\omega_3}, e_\xi\}$			Defuzzification:	weighted average
		N	**Z**	**P**	AND method:	min
Antecedent$_1$	**N**	N	N	Z	OR method:	max
$\{e_s, e_{\theta_3}, e_\psi\}$	**Z**	N	Z	P	Implication:	min
	P	Z	P	P	Aggregation:	max

Abbreviation: FLC, fuzzy logic controller.

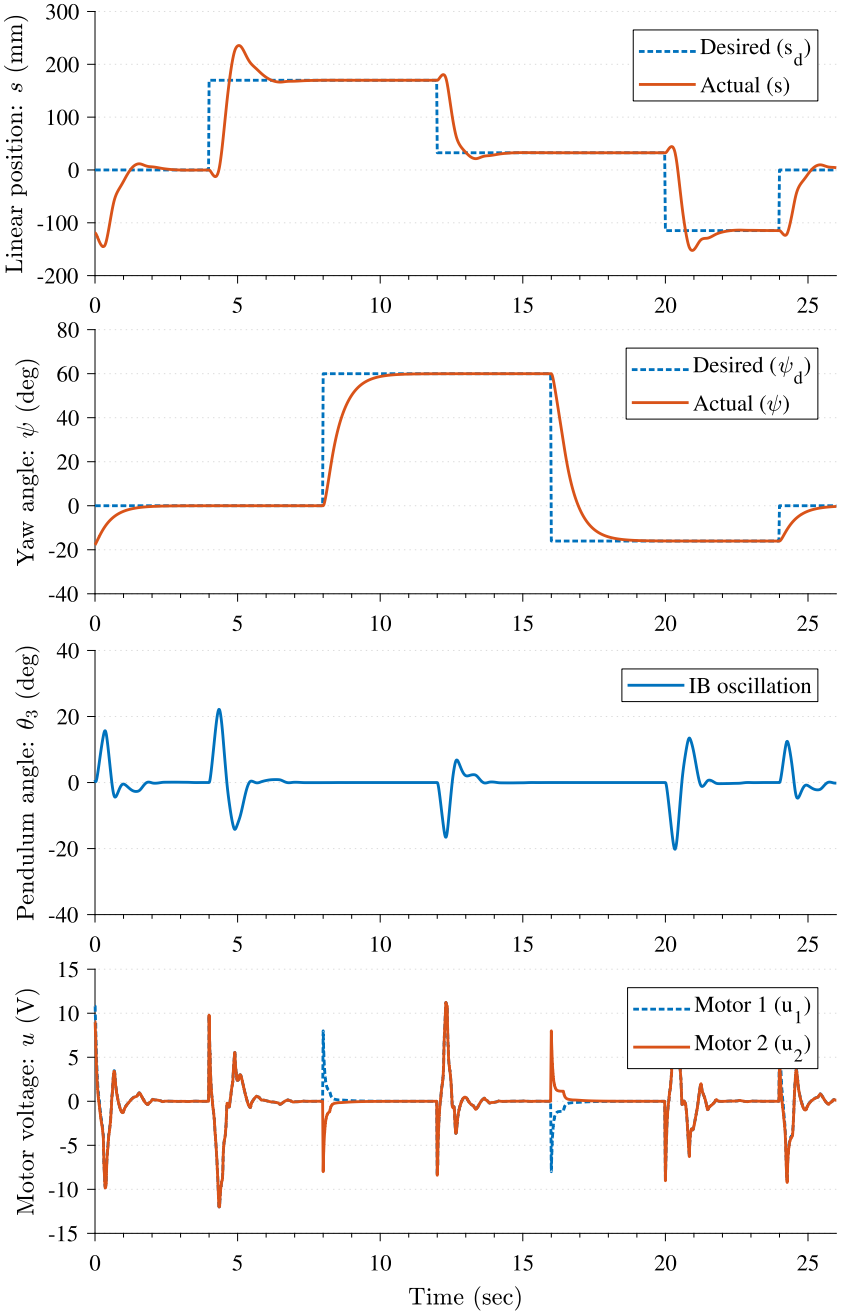


FIGURE 10 Simulation results of the closed-loop behavior

that could reduce differences and improve the control performance.

5.2 | Look-up table-based method

The behavior of an FLC is described with its fuzzy surface as it plainly expresses the relationship between the crisp inputs and outputs of the designed FLC [44,51]. As an example, the fuzzy surface of FLC2 is depicted in the first row of Figure 11, and it clearly displays the control action u_{θ_3} produced as a function of the inputs e_{θ_3} and e_{ω_3}.

The fuzzy surface can be approximated with a LUT. The LUT is generated by evaluating the possible input combinations and registering the corresponding control signal (e.g., in the case of FLC2, $e_{\theta_3} = 5$ deg and $e_{\omega_3} = 73$ deg/s input values result in a $u_{\theta_3} = 7.72$ V control signal). Consequently, each LUT element corresponds to certain input pairs. A simple MATLAB function can perform the LUT generation. In that function, the input ranges are partitioned into n_1 and n_2 evenly spaced points, and an $n_1 \times n_2$ size LUT is generated, the elements of which are determined by raking through the input ranges point-by-point and registering the corresponding control action (in equivalent PWM value d_{PWM}).

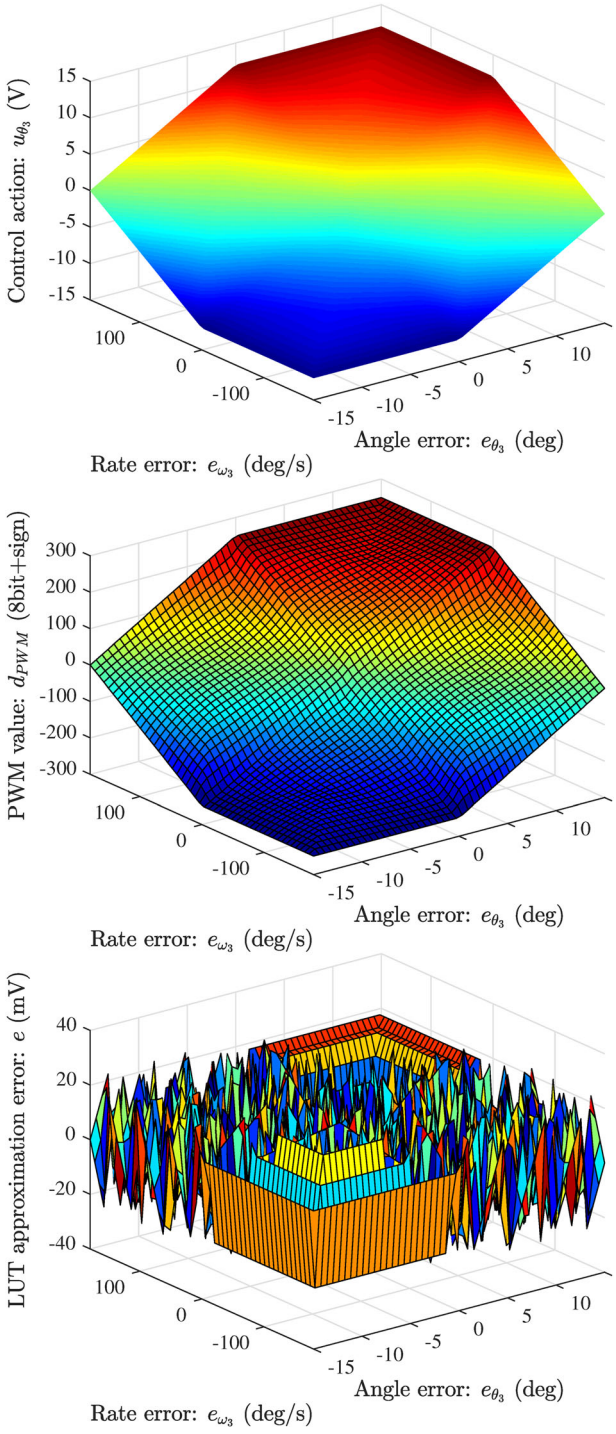


FIGURE 11 Fuzzy surface of FLC2 (top), LUT-based approximation (middle), and approximation error (bottom). FLC, fuzzy logic controller; PWM, pulse-width modulation

The generated LUT can be written into a header file, which can be easily included in the embedded code.

The generated LUT is stored in the flash memory of the MCU, and the crisp output is selected by determining the table indexes based on the input (error) signals. The available memory determines the $n_1 \times n_2$ LUT size. In the case of the Arduino Nano, the control loop relies on an ATmega328 MCU with a flash memory of 30 kilobytes. (In the default Fast PWM Mode, 8 bit resolution PWM signals are generated by the Arduino. The 8 bit duty cycle is extended with a sign bit denoting the rotational direction. Students are recommended to store the 9 bit information in integer data types).

As the control scheme (Figure 8) is based on three interconnected FLCs, three LUTs are required to be stored in the flash memory of the MCU. This means that each FLC is characterized by a LUT of $M_{\text{LUT}} = 5$ kB size (taking into account that the 15 kB program storage is kept for the rest of the implementation). Let the ranges of input variables be represented with the same number of points, that is, $n_1 = n_2$ for FLC1–3; then, $n_1 = n_2 = \sqrt{M_{\text{LUT}}/r_0} \approx 50$, where $r_o = 2$ bytes denotes the size of the crisp output elements. This 50×50 sized LUT approximation of FLC2 is displayed in the second row of Figure 11, where the crisp output is given in PWM values. The third row of Figure 11 illustrates the approximation error. The approximation can be improved by increasing both the number of input points (n_1 and n_2) and the PWM signal resolution.

The resolutions of inputs, that is, the smallest detectable incremental change of input signals, are calculated as

$$\text{res}_{i,\text{FLC}j} = \frac{R_{i,\text{FLC}j}}{n_i - 1}, \quad i = \{1, 2\}, \quad j = \{1, 2, 3\}, \tag{5}$$

where R denotes the input range (i.e., difference between the maximum and minimum values), i indicates the first or second input, and j denotes the FLC number. The resolutions of input signals of FLC1–3 considering 50×50 LUT approximations are summarized in Table 5. Given that the $R_{i,\text{FLC}j}$ input range is partitioned into n_i evenly spaced points, therefore, based on the resolution $\text{res}_{i,\text{FLC}j}$ and input range limit $R^{\min}_{i,\text{FLC}j}$, the table indexes (row: ind_1 and column: ind_2) that correspond to the instantaneous error signals are calculated as

$$\text{ind}_{i,\text{FLC}j} = \text{round}\left(\frac{e_{i,\text{FLC}j} - R^{\min}_{i,\text{FLC}j}}{\text{res}_{i,\text{FLC}j}}\right), \tag{6}$$

TABLE 5 Resolutions of input signals ($n_1 = n_2 = 50$)

FLC No.	R_1	R_2	res_1	res_2
1	0.3 (m)	0.6 (m/s)	0.0061 (m)	0.0122 (m/s)
2	30 (deg)	360 (deg/s)	0.6122 (deg)	7.3469 (deg/s)
3	90 (deg)	200 (deg/s)	1.8367 (deg)	4.0816 (deg/s)

Abbreviation: FLC, fuzzy logic controller.

where $e_{i,\mathrm{FLC}j}$ denotes the ith input signal of the jth FLC. Then, the crisp output ($u_{\mathrm{FLC}j}$) is selected employing the calculated table indexes in the corresponding LUTj:

$$u_{\mathrm{FLC}j} = \mathrm{LUT}j(\mathrm{ind}_{1,\mathrm{FLC}j}, \mathrm{ind}_{2,\mathrm{FLC}j}). \tag{7}$$

As an example, the crisp output u_s of FLC1 is calculated based on the instantaneous error signals (e_s, e_v), input resolutions ($\mathrm{res}_{1,\mathrm{FLC1}}$, $\mathrm{res}_{2,\mathrm{FLC1}}$) and input range limits ($R^{\min}_{1,\mathrm{FLC1}}$, $R^{\min}_{2,\mathrm{FLC1}}$) in the following three steps:

$$\begin{aligned} \mathrm{ind}_1 &= \mathrm{round}\left(\frac{e_s - R^{\min}_{1,\mathrm{FLC1}}}{\mathrm{res}_{1,\mathrm{FLC1}}}\right), \\ \mathrm{ind}_2 &= \mathrm{round}\left(\frac{e_v - R^{\min}_{2,\mathrm{FLC1}}}{\mathrm{res}_{2,\mathrm{FLC1}}}\right), \\ u_s &= \mathrm{LUT1}(\mathrm{ind}_1, \mathrm{ind}_2), \end{aligned} \tag{8}$$

where LUT1 denotes the 50 × 50 sized LUT approximation of FLC1, while the $R^{\min}_{1,\mathrm{FLC1}} = -0.15$ m, $\mathrm{res}_{1,\mathrm{FLC1}} = 0.0061$ m, $R^{\min}_{2,\mathrm{FLC1}} = -0.3$ m/s and $\mathrm{res}_{2,\mathrm{FLC1}} = 0.0122$ m/s constants are given in Figure 9 and Table 5.

The supplementary online material contains the Arduino listing of the LUT-based implementation of two-input one-output FLCs. This implementation technique is proposed for educational purposes, as it is simple, straightforward and requires a small number of calculations [34].

5.3 | Experimentation

Once the LUTs are generated, the LUT-based FLC algorithm is implemented, and the embedded firmware code is finalized, the experimentation on the real SBR is carried out. By this point, students have already created a minimal working environment in Arduino because both the initialization of peripherals and evaluation of sensor data has been performed in the earlier project assignments (see Section 2). Therefore, the firmware code must be extended to execute the following tasks in every t_s sampling time.

1. The instantaneous process variables s, v, θ_3, ω_3, ψ, and ξ are determined from sensor data.
2. The realization of body angle θ_3 is smoothed because the effect of external acceleration drastically decreases the reliability of pure accelerometer-based tilt angle calculations [48].
3. The error signals e_s, e_v, e_{θ_3}, e_{ω_3}, e_ψ, and e_ξ are calculated, and then the crisp outputs u_s, u_{θ_3}, and u_ψ are determined employing the LUT-based FLC algorithm.
4. The motors are excited according to Equation (4) to simultaneously guarantee the planar motion of the SBR and also to stabilize the pendulum around its unstable upright position.
5. Measurement results are sent to the PC for online monitoring and offline evaluation.

5.3.1 | Filtration

Accelerometers produce reliable orientation-related sensor data in stationary states. In this case, the decomposition of the gravity vector is used to determine the raw realization of θ_3 (pendulum angle) as $\theta_3^{\mathrm{raw}} = \arctan2(a_y, a_z)$, where a_y and a_z are the accelerometer measurements [38,48]. However, in a dynamic situation, the gravity vector cannot be distinguished from the SBR acceleration; thereby, the accuracy of θ_3^{raw} is drastically reduced. To provide reliable measurements in both states, sensor fusion techniques (e.g., Kalman or complementary filters) are applied that combine gyroscope and accelerometer data to estimate θ_3 accurately [2,9,12,26,37,45]. Due to its simplicity, the application of a complementary filter is recommended in the assignment for the purpose of smoothing θ_3^{raw}.

The complementary filter fuses the integrated gyro data and the accelerometer-based realization (θ_3^{raw}) via high-pass and low-pass filters, respectively, thereby providing an estimate $\hat{\theta}_3$ of θ_3 as [41]:

$$\hat{\theta}_3 = \frac{1}{1 + Ts}\theta_3^{\mathrm{raw}} + \frac{Ts}{1 + Ts}\frac{\Omega_3}{s}, \tag{9}$$

where T denotes the filter time constant, Ω_3 is provided by the gyroscope, and s is the Laplace Transform frequency variable. Applying the backward finite difference approximation, the discrete-time model of (9) is derived as follows:

$$\hat{\theta}_3(i) = \alpha(\theta_3(i-1) + \Omega_3(i)t_s) + (1-\alpha)\theta_3^{\mathrm{raw}}(i), \tag{10}$$

where $\alpha = \frac{T}{t_s}/(1 + \frac{T}{t_s})$ is the filter coefficient.

Students are required to both implement Equation (10) in the embedded code for smoothing the pendulum angle realization and experimentally tune the filter coefficient α.

5.3.2 | Measurement results

The control performance is analyzed via the evaluation of both the static and dynamic behaviors of the SBR. The dynamic behavior shows how effectively the SBR retains

its stability, which can be tested by pushing the robot off from its stable position.

The measurement results are depicted in Figures 12 and 13. From the top, the first and second rows compare the linear position values (instantaneous position s with desired s_d) and yaw angles (ψ with ψ_d), respectively. The third row shows the IB angles (raw realization θ_3^{raw} and estimated angle $\hat{\theta}_3$), while the last row depicts the applied voltages during the stabilization. The measurement results highlight that the proposed fuzzy control scheme ensures the stabilization of both the SBR motion and its IB. Based on the reference tracking performance shown in Figure 12, the desired positions are achieved rapidly with some overshoot in approximately 2 s, while the yaw angle stabilization is slower and requires 4 s to settle. This outcome was expected because the simulation had shown similar results (see Figure 10). Moreover, the slower response for the yaw angle compensation was caused by both the considerably larger input ranges and less intense excitation output of FLC3 (see Figure 9). The complementary filter was set up with an $\alpha = .99$ coefficient and smoothed the raw angle realization to a satisfactory degree. If there is no external perturbation, then the range of the IB angle is within $\pm 1°$, while the reference signal is tracked with ± 6 mm error. The real dynamic behavior of the SBR is displayed in Figure 13. An intense

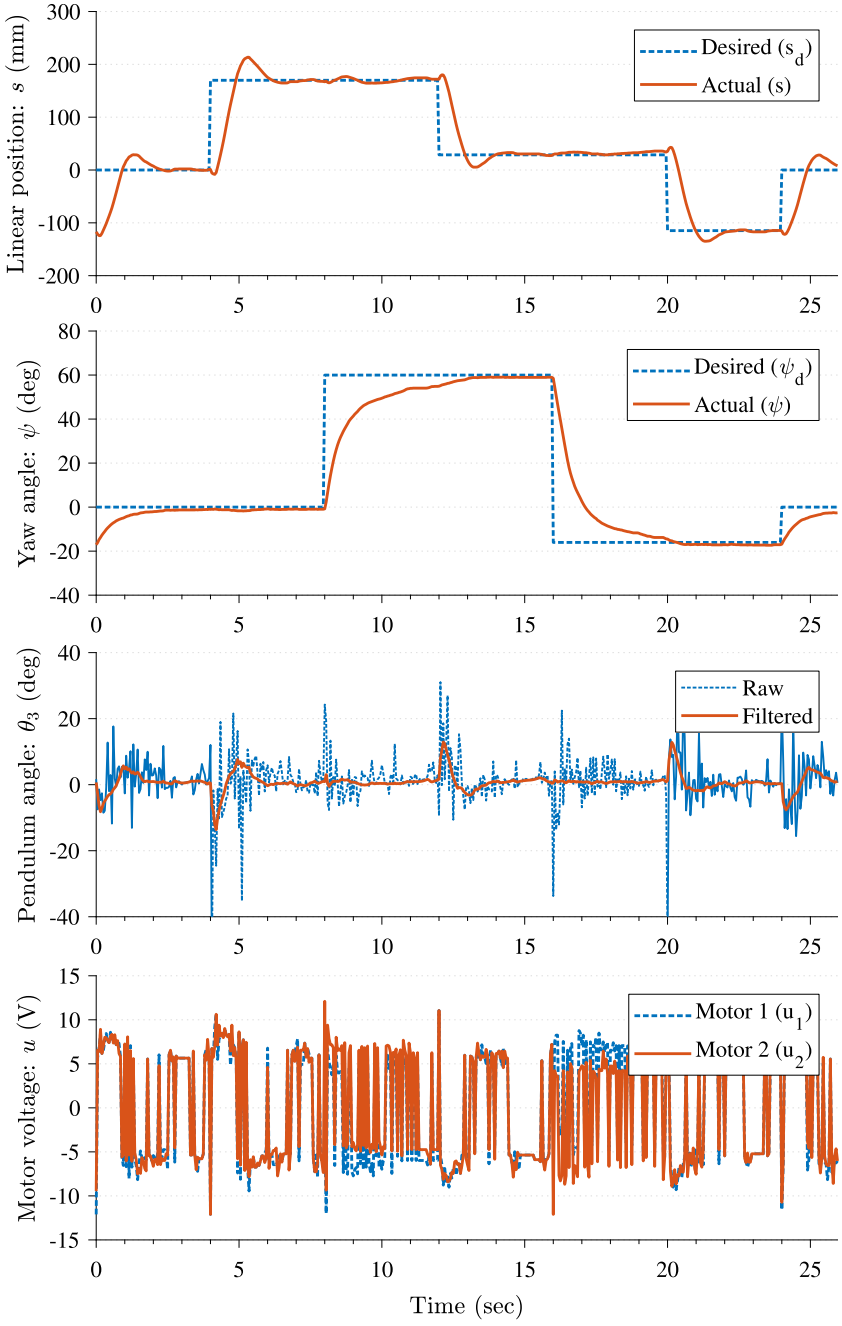


FIGURE 12 Measurement results of the closed-loop behavior: reference tracking performance (video of the closed-loop in the supplementary online material [43])

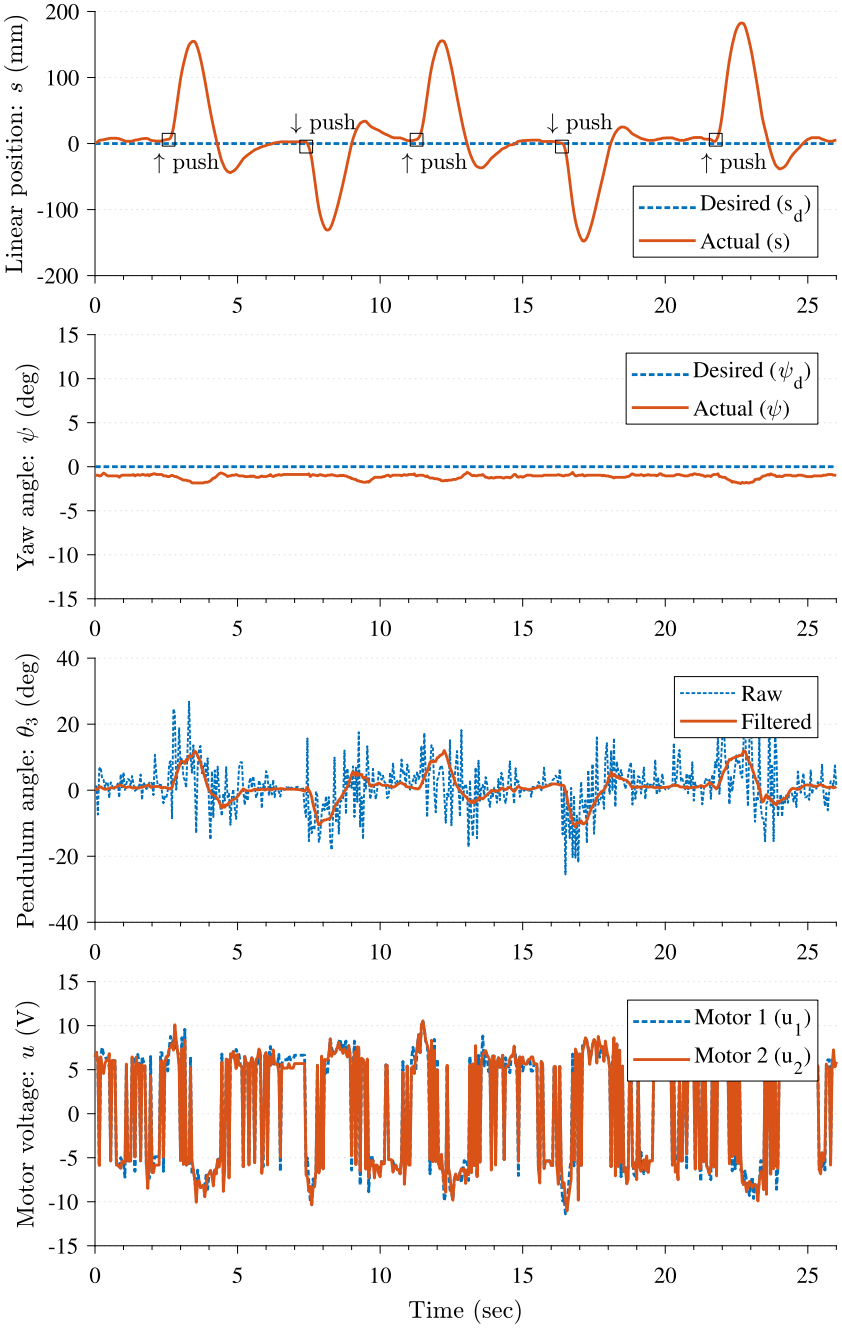


FIGURE 13 Measurement results of the closed-loop behavior: stability against external perturbations (video of the closed-loop in the supplementary online material [43])

external perturbation was applied several times (see the push events indicated with arrows) to test the stability of the closed loop system. The measurement results show that the robot was heavily pushed 150–180 mm off from its current position, and it took approximately 4 s to successfully recover to its equilibrium state. A video demonstration of the closed loop behavior is available in the supplementary online material [43].

It is obvious that the control performance is not as smooth as the simulation had shown and requires enhancements such as smaller overshoot for position control, faster dynamics for yaw angle compensation and reduced steady state errors with the application of integrator terms. Indeed, the achieved performance has been determined by different factors, for example, (a) the real SBR is characterized by strong nonlinearities (such as complex friction in the gearbox), which was not modeled and taken into account in the simulation (i.e., the mathematical model was not validated); (b) the FLCs were set up heuristically based on expert knowledge; (c) controller tuning was not performed on the real system; and (d) noisy measurements and angle estimation quality also influence the overall control performance. However, it must be borne in mind that the project requirements were fulfilled and that the assignment successfully demonstrated the aim of the project, that is, the step-by-step

development and realization of fuzzy control of a real plant. At this point, if the time constraints allow for this, the students are prompted to brainstorm and conduct experiments with different membership function shapes and distributions to enhance the control performance.

5.4 | Project objectives

The project assignment aims to (a) demonstrate the application of fuzzy logic, (b) show students that fuzzy control allows the realization of successful control that is effectively based on heuristic knowledge, and (c) dispel the illusion related to the practical implementation of FLCs by proposing a LUT-based technique that is suitable for use in education and that is also easy to code and requires minimal calculation. Students who complete the project gain experience in both the developing and embedded coding of control schemes, that is, students acquire competitive practical knowledge of sensors, actuators, signal processing and (fuzzy) control systems. It is important to highlight that the demonstrated control performances are not optimal and can be further enhanced by tuning the FLCs employing trial-and-error or optimization methods [50,51]. However, control quality enhancement is not the main goal of the project and is only recommended to advanced students who manage to achieve successful stabilization early on in the semester.

6 | ASSESSMENT AND CURRICULUM PLANS

During the academic year 2017/2018, a 2-hr trial laboratory course (registered as extracurricular activity), lasting two semesters, was held weekly to a group of 20 students at the University of Dunaújváros, Hungary. The goal of the laboratory course was to demonstrate the CSD basics with the application of Arduino systems. The course covered most of the topics discussed in this paper, that is, embedded system realization, mathematical modeling, identification and filtration, but instead of fuzzy control, the classes dealt with the application of the PID technique in embedded systems. The course attracted students and was highly successful. The authors were impressed that these students showed continuous interest throughout the semesters, increased motivation toward robotics and control, constant curiosity in deepening their knowledge, and willingness to continue the work as a part of thesis work. These very positive outcomes motivated the authors to utilize the gained initial experiences and elaborate on the proposed setup and laboratory project. The plan is to use the SBR setup in the university laboratories starting in the 2019/2020 academic year. Fifteen SBR kits, as outlined in Section 2, are going to be manufactured. The electrical components, including the Arduino, DC gear motors equipped with wheels and different breakout boards (i.e., sensors, motor drivers, and communication modules) are already in use in the laboratories. To manufacture robot frames, there are Formlabs Form 2 SLA and Ultimaker 3 professional 3D printers available at the university.

In addition to the laboratory manual, 12 educational videos of both Arduino basics and the application of the aforementioned breakout boards have been created at the university. These videos help students effectively reinforce the usage of the basic MCU peripherals, such as PWM or I2C. Each video is composed of three parts, where the first part offers a peripheral introduction and explains the examined breakout board, followed by a demonstration of the mounting of the embedded hardware parts, and, finally, the development of the MCU code is presented. Table 6 summarizes the structure of the completed educational video series.

Recommendations related to assessment and curriculum are given below. The authors anticipate that the proposed project will be of value to other university teams to design and apply both the experimental setup and the assignments in their embedded and control systems laboratories.

The successful completion of the project requires 3-hr weekly sessions. The portability of the educational tool allows the students to also work and conduct experiments outside of the university. The project is defined to be carried out by teams of three to four student members. As collaborative learning is applied where students at various performance levels work together and each member of the group affects both the effectiveness of the learning process and the success of the project, heterogeneous student groups are suggested to be formed. These heterogeneous groups are formed by taking into account the students' knowledge (academic achievement), task orientation (professional interests), social skills and abilities. The authors recommend the application of Kagan's instructions for forming the teams in a way that every team is composed of students with different knowledge levels, moreover, the student members are not close friends, nor have negative feelings toward the other members [32]. Additionally, students can also be asked about their preferences at the beginning of the first laboratory session to identify who prefers programming, hardware realization and who wishes to deal with the MATLAB/Simulink environment. Then, groups can be formed by incorporating different task orientations as well. As the enrolled student number is not large in this laboratory project, teams can be formed effectively during

TABLE 6 Structure of the educational video series

Video No.	Description	Components used in the video (beside the Arduino)
1-3	Arduino basics, peripherals, IDE, digital IO	LEDs, pushbuttons
4	Data transfer via UART communication	LEDs, Bluetooth module (HC-05)
5	Measuring voltage levels with analog inputs	Potentiometer, thermistor, accelerometer (ADXL335)
6	Emulating voltage levels with PWM outputs	LEDs
7-8	PWM-based driving of DC motors	Transistors and H-bridge modules (L293 and L298)
9	Motor speed and position measurement	H-bridge (L293) and incremental encoders
10	Measuring acceleration and rate via I2C bus	Digital IMU module (MPU6050)
11	Measuring angular rates via SPI bus	Digital gyroscope module (L3G4200D)
12	Case study: SBR and PID control	H-bridge (L298), IMU (MPU6050), and encoders

Abbreviations: IDE, integrated development environment; IMU, inertial measurement unit; PID, proportional integral derivative; PWM, pulse-width modulation; SBR, self-balancing robot; UART, universal asynchronous receiver-transmitter.

the first laboratory session. Recent studies recommend optimization methods to form heterogeneous groups if the enrolled student number is large [67].

It is recommended that the students who are to attend this laboratory course have some previous knowledge in CSD basics and embedded systems. At the beginning of the first laboratory session, the course supervisor ought to give a short presentation about the goals and main stages of the development procedure as well as demonstrate the assembled SBR and its real-time control to motivate the students. Students are frequently asked to brainstorm during the elaboration. The progress of the project work is continuously supervised by teaching assistants willing to help if any of the teams becomes stuck with solving a specific problem. At the end of the semester, each team is required to demonstrate their results in an oral presentation and submit a written project report. The final mark is given based on the fulfillment of the aforementioned requirements.

The impact and success of the laboratory project from students' perspectives can be evaluated based on questionnaires. The authors propose two different question sheets. The first such sheet (see Table 7) addresses the technical aspects and evaluates how much the project helps students understand CSD concepts. The second question sheet (see Table 8) discusses the structure of the laboratory project and experiences with the educational SBR. Student feedback will pinpoint their difficulties and help in the overall improvement of both the project and

TABLE 7 Questionnaire about the impact of the laboratory project

On a scale from 1 (not at all) to 5 (very well), rate how well the laboratory project demonstrates the following technical concepts.	
No.	**Concept/topic**
1	Embedded hardware development (usage of MCU different peripherals)
2	Sensing and actuating in an embedded environment (usage of sensors and driving motors with MCU)
3	Embedded software development (processing sensor data and coding control algorithms)
4	Mathematical modeling of dynamic systems
5	Experimentation and measurement of dynamic systems
6	Visual (grab&drop) programming in MATLAB/Simulink (simulating dynamical systems and control)
7	Feedback control concepts in theory and practice
8	Fuzzy logic and the application of FLCs
9	Development and testing of FLCs in MATLAB/Simulink
10	Realization of fuzzy control on real-world systems

TABLE 8 Questionnaire about the structure of the laboratory project

Answer the questions according to your opinion, either with text or by rating 1 (not at all) to 5 (very well).	
No.	**Question**
1	Did the assignments help you to better understand the concepts related to basic CSD and fuzzy control?
2	Did the assignments help you to acquire practical skills related to both feedback control and embedded systems?
3	How well did the assignments fit with the topics of CSD?
4	Did the laboratory sessions provide adequate time to solve the assignments? Which task required the most time?
5	Was the SBR attractive as a lab setup? Was it exciting to develop and implement CSD concepts on this plant?
6	Was it difficult to realize the hardware and software parts? How difficult was the LUT-based FLC implementation?
7	How difficult was the complete development of fuzzy control of the SBR in MATLAB/Simulink?
8	Did you have sufficient background knowledge to solve the tasks? Which topic was the most difficult?
9	How much did the educational video series and lab manual help during system development?
10	Did you find the portability of the setup useful? Was it necessary to use the university lab?

assignments at a later time. The instructors are also asked to observe the team activities and the individual involvement within the groups; moreover, they are recommended to list both the difficulties students encountered and the questions that arose during the elaboration. These reports will contribute to the increased quality of the laboratory project in the future.

Nevertheless, it is worth mentioning that the flexibility of the setup allows for further applications and developments. The setup can be easily expanded both mechanically and electrically. Mechanical extensions include the application of caster wheels or additional breadboards. The embedded electronics can be rebuilt around a faster Arduino if complex computations are performed. Moreover, the system is easily expandable with additional sensors and modules, from simple push-buttons and buzzers to a magnetometer and ultrasonic-range measurement modules. This paper has focused on fuzzy control, although, in fact, any control technique can be selected to be developed, realized and validated on the proposed setup in the CSD laboratory. The CSD concepts to be demonstrated on the setup may range from simple PID and state feedback, over Kalman filtering, to advanced control methods such as adaptive, MPC and SMC techniques. Moreover, interested students can continue their study on the SBR platform as a part of B.Sc. or M.Sc. thesis work. The proposed setup and laboratory project together contribute to teaching the embedded control systems effectively and can motivate students to obtain complex practical knowledge and thus prepare for practical work in either industry or academic fields.

7 | CONCLUSIONS

This paper proposes the application of an inexpensive, flexible, Arduino-based SBR platform in CSD laboratories and describes the complete realization procedure of a FCS as a series of laboratory assignments. The work has two main goals: on the one hand, to offer a versatile, easy-to-use, portable laboratory setup, allowing students to gain hands-on experience with embedded control systems, and on the other hand, to propose a straightforward technique in control engineering education that dispels the illusion that students usually have concerning the practical implementation of fuzzy control. The first part of the paper introduced the electromechanical structure of the proposed SBR and described the students' assignments revolving around embedded hardware and software development. Next, both the derivation and implementation in MATLAB/Simulink of mathematical models were discussed. The second part of the paper described the step-by-step development and implementation of a SBR stabilizing FCS. The authors proposed a straightforward, easy-to-code LUT-based technique for educational purposes that effectively demonstrates the practical implementation of FLCs. In the final part, the assessment and curriculum plans were described, including the methods for the evaluation of the project success and suggested future development options. The SBR platform could be offered to both undergraduate and graduate students and allows for numerous mechatronics extensions and CSD concept validation. Future work will involve the elaboration of a similar laboratory project that aims to teach the practical

implementation of linear control methods on the proposed SBR.

ACKNOWLEDGMENTS

This work was supported by the EFOP-3.6.1-16-2016-00003 project. The educational video series were made under the EFOP-3.5.1.-16-2017-00006 project. The professional 3D printers employed in the project were supported by the EFOP-4.2.1-16-2017-00011 project. The Arduino kits and robot parts, as well as the two-semester trial laboratory course, were supported by the EFOP-3.4.4.-16-0022 and NTP-FKT-17-0011 projects. Róbert Fullér was partially supported by the GINOP-2.3.4-15-2016-00003 and EFOP-3.6.1-16-2016-00010 projects. The aforementioned projects are co-financed by the European Union.

ORCID

Ákos Odry http://orcid.org/0000-0002-9554-9586

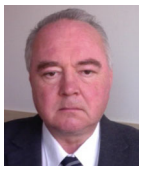
REFERENCES

1. M. Akcayol et al., *An educational tool for fuzzy logic controller and classical controllers*, Comput. Appl. Eng. Educ. **12** (2004), no. 2, 126–135.
2. A. S. Al-Fahoum and M. S. Abadir, *Design of a modified madgwick filter for quaternion-based orientation estimation using ahrs*, Int. J. Comput. Electrical Eng. **10** (2018), no. 3, 174–186.
3. N. Aliane, *A MATLAB/Simulink-based interactive module for servo systems learning*, IEEE Trans. Educ. **53** (2010), no. 2, 265–271.
4. M. Basso and G. Innocenti, *Lego-bike: A challenging robotic lab project to illustrate rapid prototyping in the mindstorms/simulink integrated platform*, Comput. Appl. Eng. Educ. **23** (2015), no. 6, 947–958.
5. M. K. Bayrakceken and A. Arisoy, *An educational setup for nonlinear control systems: Enhancing the motivation and learning in a targeted curriculum by experimental practices [focus on education]*, IEEE Control Syst. **33** (2013), no. 2, 64–81.
6. A. Behrens et al., *MATLAB meets LEGO Mindstorms–a freshman introduction course into practical engineering*, IEEE Trans. Educ. **53** (2010), no. 2, 306–317.
7. F. Candelas et al., *Experiences on using Arduino for laboratory experiments of automatic control and robotics*, IFAC-PapersOnLine **48** (2015), no. 29, 105–110.
8. R. P. M. Chan, K. A. Stol, and C. R. Halkyard, *Review of modelling and control of two-wheeled robots*, Annu. Rev. Control **37** (2013), no. 1, 89–103.
9. A. Chhotray and D. R. Parhi, *Navigational control analysis of two-wheeled self-balancing robot in an unknown terrain using back-propagation neural network integrated modified DAYANI approach*, Robotica **37** (2019), no. 8, 1346–1362.
10. J. H. Chow, G. E. Boukarim, and A. Murdoch, *Power system stabilizers as undergraduate control design projects*, IEEE Trans. Power Syst. **19** (2004), no. 1, 144–151.
11. K. Craig and F. Stolfi, *Teaching control system design through mechatronics: Academic and industrial perspectives*, Mechatronics **12** (2002), no. 2, 371–381.
12. F. Dai et al., *A two-wheeled inverted pendulum robot with friction compensation*, Mechatronics **30** (2015), 116–125.
13. M. Demirtas, Y. Altun, and A. Istanbullu, *Virtual laboratory for sliding mode and PID control of rotary inverted pendulum*, Comput. Appl. Eng. Educ. **21** (2013), no. 3, 400–409.
14. Engineering Accreditation Commission, Engineering Criteria 2000. Baltimore, MD: Accreditation Board for Engineering and Technology, 1998.
15. E. T. Enikov and G. Campa, *Mechatronic aeropendulum: Demonstration of linear and nonlinear feedback control principles with matlab/simulink real-time windows target*, IEEE Trans. Educ. **55** (2012), no. 4, 538–545.
16. K. Erenturk, *MATLAB-based GUIs for fuzzy logic controller design and applications to PMDC motor and AVR control*, Comput. Appl. Eng. Educ. **13** (2005), no. 1, 10–25.
17. D. K. Frederick and J. Chow, *Feedback control problems using MATLAB and the control system toolbox.*, Brooks/Cole Publishing Co., Belmont, CA, 1999.
18. From Paderborn to Pasadena, *From Paderborn to Pasadena*, IEEE Control Syst. **26** (2006), no. 2, 24–26.
19. I. Fürstner, L. Gogolák, and P. Sarcevic, *Development of telepresence technology during the teaching process at subotica tech*, J. Appl. Tech. Educ. Sci. **8** (2018), no. 4, 44–53.
20. M. Garduno-Aparicio et al., *A multidisciplinary industrial robot approach for teaching mechatronics-related courses*, IEEE Trans. Educ. **61** (2017), 55–62.
21. F. Grasser et al., *JOE: A mobile, inverted pendulum*, IEEE Trans. Ind. Electron. **49** (2002), no. 1, 107–114.
22. M. Gunasekaran and R. Potluri, *Low-cost undergraduate control systems experiments using microcontroller-based control of a dc motor*, IEEE Trans. Educ. **55** (2012), no. 4, 508–516.
23. Z.-Q. Guo, J.-X. Xu, and T. H. Lee, *Design and implementation of a new sliding mode controller on an underactuated wheeled inverted pendulum*, J. Franklin Inst. **351** (2014), no. 4, 2261–2282.
24. M. Gökbulut, C. Bal, and B. Dandil, *A virtual electrical drive control laboratory: Neuro-fuzzy control of induction motors*, Comput. Appl. Eng. Educ. **14** (2006), no. 3, 211–221.
25. H.-H. Huang, J.-H. Su, and C.-S. Lee, *A contest-oriented project for learning intelligent mobile robots*, IEEE Trans. Educ. **56** (2013), no. 1, 88–97.
26. C.-H. Huang, W.-J. Wang, and C.-H. Chiu, *Design and implementation of fuzzy control on a two-wheel inverted pendulum*, IEEE Trans. Ind. Electron. **58** (2011), no. 7, 2988–3001.
27. K.-S. Hwang et al., *Rapid prototyping platform for robotics applications*, IEEE Trans. Educ. **54** (2011), no. 2, 236–246.
28. D. Ibrahim and T. Alshanableh, *An undergraduate fuzzy logic control lab using a line following robot*, Comput. Appl. Eng. Educ. **19** (2011), no. 4, 639–646.
29. T. Inanc and H. Dinh, *A low-cost autonomous mobile robotics experiment: Control, vision, sonar, and Handy Board*, Comput. Appl. Eng. Educ. **20** (2012), no. 2, 203–213.
30. M. Ishikawa and I. Maruta, *Rapid prototyping for control education using Arduino and open-source technologies*, IFAC Proc. **42** (2010), no. 24, 317–321.
31. S. Jeong and T. Takahashi, *Wheeled inverted pendulum type assistant robot: Design concept and mobile control*, Intell. Serv. Robot. **1** (2008), no. 4, 313–320.

32. S. Kagan, Cooperative learning. Kagan Cooperative Learning San Juan Capistrano, CA, 1994.
33. J. Katona and A. Kovari, *A brain–computer interface project applied in computer engineering*, IEEE Trans. Educ. **59** (2016), no. 4, 319–326.
34. I. Kecskés, L. Székács, and P. Odry, Lookup table based fuzzy controller implementation in low-power microcontrollers of hexapod robot Szabad (ka)-II, 3rd Int. Conf. Workshop Mechatronics in Practice and Educ., 2015, pp. 76–81.
35. S. Khan et al., *Teaching tool for a control systems laboratory using a quadrotor as a plant in MATLAB*, IEEE Trans. Educ. **60** (2017), 249–256.
36. Y. Kim, S. Kim, and Y. Kwak, *Improving driving ability for a two-wheeled inverted-pendulum-type autonomous vehicle*, Proc. Inst. Mech. Eng., Part D: J. Auto. Eng. **220** (2006), no. 2, 165–175.
37. H. Lee and S. Jung, *Balancing and navigation control of a mobile inverted pendulum robot using sensor fusion of low cost sensors*, Mechatronics **22** (2012), no. 1, 95–105.
38. W. Li and J. Wang, *Effective adaptive Kalman filter for MEMS-IMU/magnetometers integrated attitude and heading reference systems*, J. Navig. **66** (2013), no. 1, 99–113.
39. Z. Li, C. Yang, and L. Fan, Advanced Control of Wheeled Inverted Pendulum Systems. Springer Science & Business Media, 2012.
40. S.-C. Lin and C.-C. Tsai, *Development of a self-balancing human transportation vehicle for the teaching of feedback control*, IEEE Trans. Educ. **52** (2009), no. 1, 157–168.
41. R. Mahony, T. Hamel, and J.-M. Pflimlin, Complementary filter design on the special orthogonal group SO (3), Proc. 44th IEEE Conf. Decision and Control, 2005, pp. 1477–1484.
42. K. Nagai, *Learning while doing: Practical robotics education*, IEEE Robot. Autom. Mag. **8** (2001), no. 2, 39–43.
43. Á. Odry, Self-balancing robot supplementary material, (2019). Available at www.appl-dsp.com; http://appl-dsp.com/self-balancing-robot-kit/
44. Á. Odry et al., Fuzzy control of a two-wheeled mobile pendulum system, IEEE 11th Int. Symp. in Appl. Comput. Intell. Inform. (SACI), 2016, pp. 99–104.
45. Á. Odry, E. Burkus, and P. Odry, LQG control of a two-wheeled mobile pendulum system, 4th Int. Conf. Intell. Syst. Appl. (INTELLI), 2015, pp. 105–112.
46. Á. Odry, J. Fodor, and P. Odry, *Stabilization of a two-wheeled mobile pendulum system using LQG and fuzzy control techniques*, Int. J. Adv. Intell. Syst. **9** (2016), no. 1,2, 223–232.
47. Á. Odry and R. Fullér, Comparison of optimized PID and fuzzy control strategies on a mobile pendulum robot, IEEE 12th Int. Symp. in Appl. Comput. Intell. Inform. (SACI), 2018, pp. 207–212.
48. Á. Odry et al., *Kalman filter for mobile-robot attitude estimation: Novel optimized and adaptive solutions*, Mech. Syst. Signal Process. **110** (2018), 569–589.
49. Á. Odry et al., Design, realization and modeling of a two-wheeled mobile pendulum system, 14th Int. Conf. Instrum. Meas. Circuits Syst. (IMCAS), 2015, pp. 75–79.
50. Á. Odry et al., *Optimized fuzzy control of a two-wheeled mobile pendulum system*, Int. J. Control Syst. Robot. **2** (2017), 73–79.
51. Á. Odry et al., *Protective fuzzy control of a two-wheeled mobile pendulum robot: Design and optimization*, WSEAS Trans. Syst. Control **12** (2017), 297–306.
52. H. M. Omar, *Enhancing automatic control learning through Arduino-based projects*, Eur. J. Eng. Educ. **43** (2017), 1–12.
53. K. Pathak, J. Franch, and S. K. Agrawal, *Velocity and position control of a wheeled inverted pendulum by partial feedback linearization*, IEEE Trans. Robot. **21** (2005), no. 3, 505–513.
54. R.-E. Precup et al., *Experiment-based teaching in advanced control engineering*, IEEE Trans. Educ. **54** (2011), no. 3, 345–355.
55. G. V. Raffo et al., *Two-wheeled self-balanced pendulum workspace improvement via underactuated robust nonlinear control*, Control. Eng. Pract. **44** (2015), 231–242.
56. M. Rampazzo, A. Cervato, and A. Beghi, *Remote refrigeration system experiments for control engineering education*, Comput. Appl. Eng. Educ. **25** (2017), no. 3, 430–440.
57. R. M. Reck, *Common learning objectives for undergraduate control systems laboratories*, IEEE Trans. Educ. **60** (2017), 257–264.
58. A. Rmilah et al., *A PC-based simulation platform for a quadcopter system with self-tuning fuzzy PID controllers*, Comput. Appl. Eng. Educ. **24** (2016), no. 6, 934–950.
59. Segway Inc., Segway PT, (2019). Available at www.segway.com; http://www.segway.com/
60. P. Shakouri, A. Ordys, and G. Collier, *Teaching model predictive control algorithm using starter kit robot*, Eng. Educ. **8** (2013), no. 2, 30–43.
61. P. S. Shiakolas et al., *Magnetic levitation hardware-in-the-loop and MATLAB-based experiments for reinforcement of neural network control concepts*, IEEE Trans. Educ. **47** (2004), no. 1, 33–41.
62. M. Stefanovic et al., *A LabVIEW-based remote laboratory experiments for control engineering education*, Comput. Appl. Eng. Educ. **19** (2011), no. 3, 538–549.
63. A. Steinhauser et al., *Low-cost carry-home mobile platforms for project-based evaluation of control theory*, IFAC-PapersOnLine **50** (2017), no. 1, 9138–9143.
64. J. Sun and Z. Li, *Development and implementation of a wheeled inverted pendulum vehicle using adaptive neural control with extreme learning machines*, Cogn. Comput. **7** (2015), no. 6, 740–752.
65. J. Sánchez et al., *A Java/Matlab-based environment for remote control system laboratories: Illustrated with an inverted pendulum*, IEEE Trans. Educ. **47** (2004), no. 3, 321–329.
66. A. Takacs et al., *Teacher's kit: Development, usability, and communities of modular robotic kits for classroom education*, IEEE Robot. Autom. Mag. **23** (2016), no. 2, 30–39.
67. Đ. Takači et al., *Efficiency of using VNS algorithm for forming heterogeneous groups for CSCL learning*, Comput. Educ. **109** (2017), 98–108.
68. L.-X. Wang, A course in fuzzy systems, Prentice-Hall Press, USA, 1999.
69. J.-X. Xu, Z.-Q. Guo, and T. H. Lee, *Design and implementation of a Takagi–Sugeno-type fuzzy logic controller on a two-wheeled mobile robot*, IEEE Trans. Ind. Electron. **60** (2013), no. 12, 5717–5728.
70. J.-X. Xu, Z.-Q. Guo, and T. H. Lee, *Design and implementation of integral sliding-mode control on an underactuated two-wheeled mobile robot*, IEEE Trans. Ind. Electron. **61** (2014), no. 7, 3671–3681.
71. H. Ying, Fuzzy Control and Modeling: Analytical Foundations and Applications. Wiley-IEEE Press, 2000.
72. D. F. Zapata García et al., *QuadLab - a project-based learning toolkit for automation and robotics engineering education*, J. Intell. Robot. Syst. **81** (2016), no. 1, 97–116.

AUTHOR BIOGRAPHIES

Ákos Odry is a PhD student at the Doctoral School of Applied Informatics and Applied Mathematics, Óbuda University, and an assistant lecturer in the Department of Control Engineering and Information Technology, University of Dunaújváros. He received his BSc and MSc degrees in electrical engineering from Budapest University of Technology and Economics in 2012 and 2015, respectively. His research interests include nonlinear control systems, robotics, engineering education, soft computing and adaptive control methods.



Róbert Fullér is a Professor in the Department of Informatics, Széchenyi István University. He graduated from Eötvös Loránd University, Budapest, in 1982 and received his Ph.D. from Moscow State University in 1988 and a Doctor of Science degree from the Hungarian Academy of Sciences in 2015. He was a postdoctoral fellow in the Department of Operations Research, RWTH Aachen, Germany (1990-91), a Donner Visiting Professor (1995-96) and Finland Distinguished Professor (2008-2011) at the Institute for Advanced Management Systems Research, Abo Akademi University, Finland. His current research interests include neural fuzzy systems, fuzzy multiple criteria decision making and approximate reasoning for optimization and control. He is the author of 6 books and has received more than 7000 citations.

Imre J. Rudas is a full Professor and the Head of the Steering Committee of the University Research, Innovation and Service Center. He received his MSc in Mathematics from the Eötvös Loránd University, Budapest, a Ph.D. in Robotics from the Hungarian Academy of Sciences in 1987, and a Doctor of Science degree from the Hungarian Academy of Sciences in 2004. He received a Doctor Honoris Causa degree from the Technical University of Košice, Slovakia, from "Polytechnica" University of Timisoara, Romania, from Óbuda University, and from the Slovak University of Technology in Bratislava. He was awarded with an Honorary Professor title in 2013 and an Ambassador title in 2015 by the Wroclaw University of Technology. His research topics are in robotics and control engineering. He has published more than 800 papers and received more than 3500 citations.

Péter Odry is a Professor in the Department of Control Engineering and Information Technology, University of Dunaújváros. He received his MSc and PhD degrees in electrical engineering from the University of Belgrade in 1986 and 1992, respectively. His research interests include impedance tomography, legged robotic systems, computer vision, robust control techniques and soft computing methods.

How to cite this article: Odry Á, Fullér R, Rudas IJ, Odry P. Fuzzy control of self-balancing robots: A control laboratory project. *Comput Appl Eng Educ.* 2020;1–24.
https://doi.org/10.1002/cae.22219

APPENDIX: SYSTEM DYNAMICS

This section describes both the Lagrangian $\mathcal{L}(q)$ of the system and the torque difference $(\tau_a - \tau_f)$ in Equation (2).

$$\begin{aligned}\mathcal{L}(q) = \sum_{i=1}^{2}\Bigg(&\frac{6m_w r^2 + m_b r^2}{8} + \frac{m_b l^2 r^2 \sin^2\theta_3 + J_B r^2}{2d^2} \\ &+ \frac{k^2 J_M}{2}\Bigg)\dot{\theta}_i^2 + \left(\frac{m_b l^2 + J_A}{2} + k^2 J_M\right)\dot{\theta}_3^2 \\ &+ \sum_{i=1}^{2}\left(\frac{m_b l r \cos\theta_3}{2} - k^2 J_M\right)\dot{\theta}_3\dot{\theta}_i \\ &+ \left(\frac{m_b r^2}{4} - \frac{m_b l^2 r^2 \sin^2\theta_3 + J_B r^2}{d^2}\right)\dot{\theta}_1\dot{\theta}_2 \\ &- 2m_w g r - m_b g\left(r + l\cos\theta_3\right).\end{aligned} \tag{11}$$

$$\tau_a - \tau_f = k\begin{bmatrix}1 & 0\\ 0 & 1\\ -1 & -1\end{bmatrix}\tau_M - \begin{bmatrix}f_M + f_w & 0 & -f_M\\ 0 & f_M + f_w & -f_M\\ -f_M & -f_M & 2f_M\end{bmatrix}\dot{q}. \tag{12}$$