Received December 29, 2020, accepted February 10, 2021, date of publication February 15, 2021, date of current version February 25, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3059482

A Stochastic Logic-Based Fuzzy Logic Controller: First Experimental Results of a Novel Architecture

ÁKOS ODRY, VLADIMIR TADIC, AND PETER ODRY
Department of Control Engineering and Information Technology, University of Dunaújváros, 2400 Dunaújváros, Hungary
Corresponding author: Ákos Odry (odrya@uniduna.hu)

This work was supported in part by the European Union under Grant EFOP-3.6.1-16-2016-00003 project. The work of Ákos Odry and Peter Odry was supported in part by under Grant 2020-4.1.1-TKP2020 program.

ABSTRACT In stochastic computing (SC) systems numbers are represented with mean values of random binary sequences. This paper introduces a novel fuzzy inference architecture, in which the computational mechanism is based on stochastic logic (SL). First, the basic concept of SL is described, then the architecture of the SL-based fuzzy logic controller (SFLC) is built up systematically using the derived stochastic elements. The second part of the paper demonstrates the application of the proposed techniques, where the SFLC-based control performance is evaluated on a real mechatronic system. The results show that the SL-based approach provides effective and robust control performance, simple architecture and high noise tolerance. The proposed method is also benchmarked against conventional FLCs indicating that the robustness of the stochastic architecture allowed to outperform the benchmark controllers in noisy environments.

INDEX TERMS Stochastic logic, fuzzy logic controller, fuzzy hardware, self-balancing robot.

I. INTRODUCTION

A. RELATED WORK

SC makes the hardware architecture less complex, provides simple components for arithmetic operations and enables the execution of big number of parallel computations [1], [2]. The computing robustness, fault tolerant nature, scalability and reduced consumption footprint are among the key characteristics that made this fruitful technology become popular in recent research works. The investigations aim to develop effective SC-based architectures that can be beneficially applied in image processing algorithms [3]–[6], general purpose digital filter structures [7]–[10], error correction hardware solutions [11], and artificial neural networks (ANNs) [12]. The cost of the aforementioned attributes is a trade-off between precision and latency in signal representations, since the longer the processed bit stream the higher precision is achieved. For the relaxation of this trade-off novel processing methods are proposed in the literature, which aim to both exploit the advantageous features and overcome the difficulties of this computing paradigm [13]–[15].

The fundamental concept of the application of random binary sequences belongs to Brian R. Gaines [16]–[18]. Gaines introduced the stochastic computer which uses the probability of switching a digital circuit as an analog quantity. In the proposed concept different representations were described, and the basic stochastic computing elements (inverter, multiplier, summer and integrator) were built by the help of standard logic elements. The stochastic computer was proposed as a low-cost computational system that can be used in applications where high speed (and inherently high resolution) is not required. However, nowadays technology with the tendency from MHz to GHz clock rates of electronic circuits makes the concept possible to provide both accurate and robust computations.

ANNs are one of the key areas that have been inspired by SL [19]. The massive parallelism that can be achieved by simple computational elements with reduced hardware complexity is among the important issues that SL was applied for in computer science [20]. The information processing of SL-based ANNs, moreover, both the architectural and functional features of ANN design have already been studied in the earlier years of SC in reference [21]. Investigations were continued in [22], where SC techniques were used for implementing large parallel ANNs on field-programmable gate arrays (FPGAs). Reference [23] also recommended a concept to design random-pulse ANNs whose architecture is well suited for very large-scale integration (VLSI). In [24]

The associate editor coordinating the review of this manuscript and approving it for publication was Moussa Boukhnifer.

several state machine-based computational elements were elaborated to perform both linear gain and exponentiation functions by the use of stochastic binary signals. Recently, massive attention was given to Deep Convolutional Neural Networks (DCNNs) [25], [26]. References [27], [28] proposed advantageous hardware implementations, which provide both optimized hardware footprint (i.e., minimized area and power consumption) and high network accuracy. One of the industrial applications was presented in [29], where both SC and ANNs were used to control an FPGA-based induction motor drive system. The work showed that SL enhances the arithmetic operation of FPGAs, requires less resources, and provides easy implementation of both NNs and classical algorithms on a single low-complexity FPGA. Similar FPGA-based implementation results were proposed in [30], where SC aided feed-forward ANNs were utilized for the control of wind turbine systems. Moreover, references [31], [32] proposed novel approaches with enhanced SC-based arithmetic operations for prediction, function fitting and machine translation problems, where noise-tolerant ANNs were derived with both reduced hardware constraints and improved accuracy compared to binary designs. The proposed solution even outperformed the binary counterpart, when significant noise disturbed the computation. Several recent works have addressed the efficient implementation of ANNs with the aim to provide highly parallelized accelerators that are characterized by reduced consumption and latency along with improved accuracy, thereby enabling the efficient implementation in embedded systems. Recent solutions propose new computing methods [33], different coding schemes [34], efficient arithmetic units and simplified architectures [35], [36]. Comprehensive overviews of recent solutions are provided in [37].

The generation of statistically independent binary sequences with arbitrary ratios of 1's to 0's using simple digital circuits was investigated in the early years of SC [38]. In [39] different approaches were described for both analog to digital (AD) and digital to analog conversions (DA). These techniques were used in data acquisition circuits of meteorological supervision centers, where the random binary sequences were processed in VLSI circuits. A space-efficient (i.e., few AND, OR, NOT gates and flip-flops) fully parallel SC architecture was proposed in [40] for large number of arithmetic calculations; the architecture provided high efficiency in terms of consumed area and delays. An extended set of SC-based arithmetic operations was proposed in [41], moreover, the interfaces between analog and stochastic domains were addressed. SL has also been investigated by the instrumentation and measurement community. A stochastic instrument for true root mean square (RMS) measurements was presented in [42], where 0.1% full scale inaccuracy was demonstrated. References [43]–[45] proposed SC-based iterative decoding architectures. It was shown that the stochastic decoder outperformed the conventional analogue decoders by its high operational speed, low power consumption, technological independency and near-optimal performance based on simulation results for low-density parity-check (LDPC) codes. In reference [46] the effectiveness and fault tolerance of SL-based reconfigurable architectures implemented for processing operations on a datapath were studied, moreover, the error sources were also analyzed. Reference [11] both provides a great overview of SC-based accelerators and describes important design guidelines for development of energy-optimized hardware architectures.

The efficient hardware implementation of FLCs was also studied and different fuzzy-logic chips were proposed using both analog and mixed-digital circuits [47]–[51]. In references [47], [48] FLCs were designed using CMOS technology. Simple SL-based digital architectures with short response time were designed for the implementation of arithmetic functions in the deffuzification process. A SL-based center-of-gravity defuzzifier circuit was proposed, where both the firing strength of rules and singleton output variables were represented with random sequences. A stochastic architecture was described in [49], where the inference process was also represented with binary bit streams, moreover, the SL-based fuzzy operators (i.e., min and max operators) were introduced as well. A possible implementation of FLCs for FPGAs was described in [50]. In the proposed prototype architecture both look-up tables and parallel-serial converters were utilized for producing random binary bit streams; the defuzzification process was realized with accumulators, serial-parallel converters and a divider algorithm. A SL-based fuzzy inference method was described in [51]. The authors proposed a novel fuzzification method, i.e., new conditions were described for the evaluation of inputs for representing the membership values with random binary bit streams. References [52], [53] provide detailed overviews of different fuzzy hardware solutions. Moreover, an excellent review of SL-based applications is given [54].

B. CONTRIBUTION OF THE PAPER

The aforementioned papers highlight that SL provides a new perspective to realize complex computational architectures with simple hardware elements. Nowadays technology, i.e., GHz frequency ranges, high integration and low power solutions, provides the opportunity to both design and apply control techniques that consist of simple logic gates. As a result, this article both combines different methods introduced in the literature [17], [18], [39], [49], [51] and constructs a novel SFLC architecture which consists of simple logic gates, counters and comparators. Furthermore, a comprehensive experimental validation is outlined, where the proposed architecture is utilized for the stabilization of a real self-balancing robot (SBR). Namely, we apply the stochastic architecture and design a two-loop SL-based fuzzy control scheme for the plant. Then, the performance is both evaluated and compared with the conventional FLC-based counterpart. In this comparative analysis, the fuzzy parameters (i.e., membership functions, rules, ranges and defuzzification) were identical at both control methods.

The experimental results highlighted that the SFLCs provide more robust performance than their conventional counterparts, especially when noisy measurements are present in the control loop. Therefore, the proposed SFLC architecture is characterized by both high noise tolerance and flexible design. Moreover, it can easily be interfaced in embedded systems as either a simple program code or fuzzy hardware accelerating engine consisting of analog, digital or mixed-signal circuits. The recommended architecture is efficiently applicable in FPGAs, where the control circuit can be made adoptable anytime to the application. As to the authors' best understanding, the performances of similar SFLC architectures were evaluated only with simulation results [47]–[51]. Therefore, this paper aims to demonstrate both a novel SFLC architecture and its first implementation results on a real mechatronics system. Moreover, the proposed full architecture of stochastic FLCs has not been published yet. The paper presents the robustness of the SFLC-based architecture with both simulation and experimental results.

The remainder of the paper is organized as follows. In section II a brief summary of SC, SC elements and fundamental methods is given. Section III deals with the fuzzy logic related stochastic elements. In section IV the applied SBR and its mathematical model are described. The reference fuzzy control scheme is discussed in section V. Then, beginning with section VI the design of the equivalent SFLC architecture is presented. Sections VII and VIII highlight the achieved control performances and experimental results, while in section IX our conclusions and recommendations for future studies are given.

II. BASIC CONCEPT

SC systems are composed of three parts. First, the input interface performs the conversion from conventional deterministic signal representation domain to stochastic representation. Then, the stochastic processing system executes the arithmetic operations on stochastic sequences. Finally, the output interface converts back the calculation results from stochastic domain to deterministic signal representation domain. This section gives a brief summary about the concept of SC, SC elements, and techniques employed for arithmetic operations. The summary focuses only on the standard SL solutions, since those will be used in the formulation of SFLCs later.

A. STOCHASTIC REPRESENTATION

In SL, signals are represented with random binary bit streams (stochastic bit streams) in which the information is codified in the probability of any given bit is being 1 in the stochastic sequence. Therefore, a random binary bit stream is a Bernoulli sequence, a sequence of randomly varying ones (1's) and zeroes (0's), where the probability of a bit being 1 is independent of any previous bits. This stochastic representation enables to perform complex arithmetic operations by using simple digital circuits (i.e., combination of AND and OR gates) on binary bit streams.

Random binary sequences are produced by the help of a comparator and random number generator (see the left side of Fig. 1) [17], [18], [55]. The codification procedure compares the input signal x to a random number r generated in every clock cycle. In this manner, if the input signal $x \in [0, 1]$ is compared to a uniform random number $r \in [0, 1]$, then the generating probability p that a pulse is produced at the output of the comparator is simply equal to the input signal, i.e., $p = x$ where $P(X_i = 1) = p$, and $i = 1, \ldots, N$. Thus, the signal value is represented with the generating probability. This is the unipolar representation of real numbers, where the analysis is restricted to signals in range [0, 1]. The other common representation is the bipolar encoding format, which also enables to represent the negative numbers, i.e., the real number is $x \in [-1, 1]$ and the generating probability is $P(X_i = 1) = (x + 1)/2$. Larger ranges can also be encoded in SC with the application of scaling functions. Hereinafter, we focus our analyses to unipolar arithmetic operations, since those are employed in the SFLC architecture.

Once the stochastic representation is available, it gives only an approximation of the input signal x. The estimation (or approximation) $\hat{x}$ of the original signal is given as the relative frequency of 1's in a sufficiently long bit stream. Let X_i and N denote the ith element and length of the sequence, respectively, then the estimation is given as

$$\hat{x} = \frac{1}{N} \sum_{i=1}^{n} X_i. \tag{1}$$

As Eq. (1) indicates, the statistical average of the binary stream gives the estimation of the signal x (i.e., the estimation of the generating probability, since $\hat{p} = \hat{x}$), thus the longer stream is processed, the better accuracy is obtained ($\hat{x} = x$ only if $N \to \infty$). The accuracy of estimation is defined by its variance; it is considered as a coding noise that decreases by time [3], [17], [18]:

$$\mathrm{Exp}\left(\hat{x}\right) = x, \quad \mathrm{Var}\left(\hat{x}\right) = \frac{x\,(x-1)}{N}. \tag{2}$$

The aforementioned inverse-proportional relationship results in that the precision is directly influenced by the length N of stochastic sequences in SC systems. However, processing longer bit-streams results in longer data conversion time (i.e., a trade-off between precision and latency), thus higher clock rates are required to achieve satisfying performance.

B. RANDOM NUMBER GENERATION

Linear feedback shift registers (LFSR) are used most commonly for random number generation in digital domain. The LFSR is a shift register whose most significant bit is calculated as a XOR function of a certain set of bits. The n-order (flip-flop based) shift register produces $2^n - 1$ long sequences of pseudo-random numbers [18], [39]. The LFSR constitutes the most basic approach, however the literature also proposes more advanced methods to generate high accuracy bit streams that represent random numbers with the application of mixed

analog-digital noise sources in feedback configurations [18], [56]–[58].

C. ARITHMETIC OPERATIONS

Arithmetic and logic operations (e.g., multiplication and addition) can easily be performed on stochastic bit streams by the help of simple logic gates. We limit our discussion to those arithmetic units that are relevant in our study. A complete analyses is provided in [59].

1) MULTIPLICATION AND DIVISION

Multiplication is the most common arithmetic operation. Let X_1 and X_2 denote the input bit streams that encode the deterministic values x_1 and x_2 in stochastic domain, moreover, let Y denote the output bit stream, i.e., $P(Y=1)=y$. If p_1 and p_2 represent the corresponding generating probabilities of the two uncorrelated input stochastic sequences, then a simple AND gate executes the multiplication, since

$$\begin{aligned} y &= P(X_1=1 \wedge X_2=1) \\ &= P(X_1=1) \wedge P(X_2=1) = p_1 p_2. \end{aligned} \tag{3}$$

Therefore, the probability of being high level at the output of the AND gate equals to $p_1\ p_2$, i.e., the product of the input signals is both encoded and realized.

Division is a more difficult to implement in stochastic domain due to the inherent nonlinearity and range maintenance issues. The approximated division is realized with a simple JK flip-flop, namely, at output of the flip-flop $p_1/(p_1-p_2)$ is provided in stochastic domain. More advanced divider circuits are described in [16], [41], [60].

2) ADDITION AND SUBTRACTION

Stochastic summation constitutes the other elementary operation. A simple OR gate produces probabilistic addition, i.e., the generating probability at the output of OR gates equals to

$$\begin{aligned} P(Y=1) &= P(X_1=1 \vee X_2=1) \\ &= p_1 + p_2 - p_1 p_2. \end{aligned} \tag{4}$$

The addition p_1+p_2 can be approximated with an OR gate only if the pulse densities are kept very low (i.e., $p_1\ p_2 \ll p_1+p_2$). The extra $p_1\ p_2$ term in Eq. (4) can be compensated for with a multiplexer (MUX), which performs scaled addition. Let p_3 denote the probability of the select line of MUX, then the output stream is

$$\begin{aligned} P(Y=1) &= P(X_1=1 \wedge X_3=1) \vee P(X_2=1 \wedge X_3=0) \\ &= p_1 p_3 + p_2(1-p_3). \end{aligned} \tag{5}$$

Therefore, the scaled addition with factor of 2 is realized if the select input is driven with a stochastic bit stream of $p_3=0.5$ generating probability. As a result $(p_1+p_2)/2$ is executed with the MUX.

It is worth mentioning that the XOR gate is also employed for similar arithmetic operations [13]. Namely, if the input stochastic bit streams X_1 and X_2 are unequal, then the XOR gate produces 1 at its output, i.e.:

$$\begin{aligned} P(Y=1) &= P(X_1=1 \wedge X_2=0) \vee P(X_1=0 \wedge X_2=1) \\ &= p_1 + p_2 - 2p_1 p_2. \end{aligned} \tag{6}$$

Similarly, the NOT gate (inverter) provides $1-p$ if the input stream x is characterized by generating probability p. Various approximating circuits have been designed which execute more accurate addition and subtraction operations, see the references [12], [16], [37], [40] for more detail.

3) COMPARISON

The comparison of signals in stochastic domain can also be performed with simple logic gates. Namely, if the stochastic representations are generated by a shared random number generator, then their comparison can be realized in stochastic domain. Namely, the stochastic sequences shall be produced by comparing the input signals to the same random number. The minimum operation (selecting the least of the available signals) is performed by a simple AND gate. The maximum operation (selecting the biggest of the available signals) is executed by a simple OR gate [49].

D. STOCHASTIC NUMBER CONVERSION TO DETERMINISTIC DOMAIN

The third part of the SC system is the output interface which converts the results obtained in stochastic domain back to deterministic representation. This conventional (crisp) representation can be obtained by estimating the mean value of a sufficiently long stochastic sequence. Keeping in mind, that the information is codified in the number of 1's in a stochastic sequence, the conversion can be performed by a simple counter of $N+1$ states, which estimates the generating probability of input stream (i.e., every state represents a numerical value). Additionally, moving-average converter circuits can be used to generate continuously updated digital average of stochastic bit streams [23].

Based on Eq. (1), an n-bit counter accumulates the input stream X for 2^n clock cycles, i.e., $N=2^n$. Moreover, the accumulated value $s=\sum X_i$ represents the number of non-zero bits in the stochastic bit stream, which can be used directly to estimate the deterministic value as $\hat{x}=s/2^n$. Due to this accumulation process, the estimate $\hat{x}$ is updated with a period of 2^n clock cycles. The up/down counter is incremented by a unit count if at a clock pulse its inputs are INC=1 and DEC=0. Similarly, the counter is decremented by a unit count if INC=0 and DEC=1. The counter remains unchanged if its inputs are on the same level (INC=DEC). In the time domain, the expected value $\hat{s}$ of the counter can be given with Eq. (7) if the input lines are driven from a pair of Bernoulli sequences [18]:

$$\hat{s} \approx s(0) + \frac{1}{NT}\int_0^t w(\tau) - e(\tau)\, d\tau, \tag{7}$$

where $s(0)$ is the initial value of the counter, T is the clock interval in seconds, N represents the number of states of

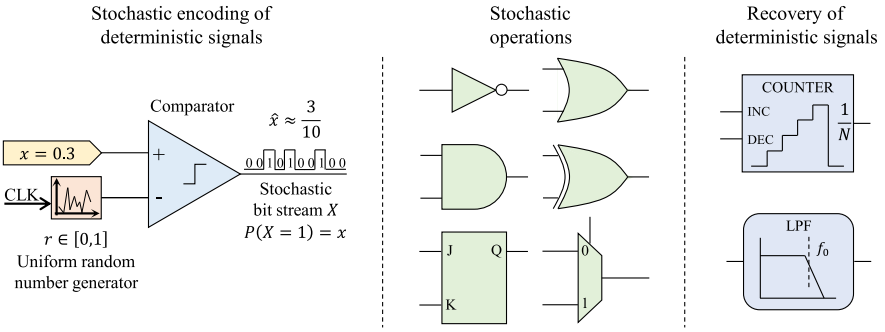


FIGURE 1. The basic structure of SC systems.

the counter, while w and e denote the probabilities that the counter will be incremented or decremented, respectively. Therefore, the counter performs integration with gain $1/NT$.

On the other hand, the conversion of stochastic numbers to deterministic representation can also be performed with low pass filters (LPFs) [39], [59]. This way the DC component of the stochastic sequence is obtained by standard filter circuits. The DC component represents the information the bit stream codifies. Different filter characteristics and filter design conditions have been addressed in [39]. The LPF-based conversion enables to update the estimate of deterministic value in every clock cycle. The simplest LPF structure estimates the generating probability from the fast stochastic bit stream X as:

$$\hat{x}_{i+1} = \hat{x}_i (1 - \beta) + \beta X_i, \tag{8}$$

where X_i denotes the stochastic stream value in the ith epoch and β defines the step size. The step size represents the cut-off frequency f_0 of the filter; it controls the estimation convergence, i.e., small β results in slow estimation with better accuracy and stability.

Fig. 1 summarizes both the basic structure of SC systems and most fundamental SC components. This section pointed out, that enormously simplified and space efficient hardware circuits (or software implementations) can be achieved with stochastic representation. Furthermore, SC is characterized by strong error tolerance, e.g., bit-flip errors cause relatively small differences in a sufficiently long stochastic bit stream, since the information is codified in the mean value of a stochastic sequence. However, the obtained resolution is determined by the clock frequency employed to generate the stochastic representations, therefore the clock rate has to be much higher than the dynamics of the controlled system.

III. STOCHASTIC FUZZY COMPONENTS

This section introduces the constituent parts of the proposed SFLC architecture. The stochastic representation of the membership function values is obtained by the method described in [51]. Once stochastic sequences are generated, the conventional logic gates introduced in section II are used to perform both the inference mechanism and aggregation of fuzzy

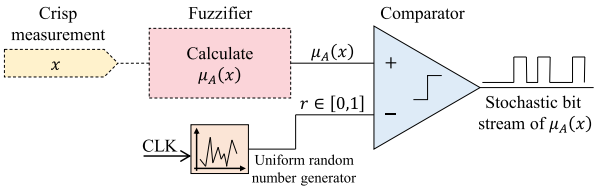


FIGURE 2. Coding circuit that generates the stochastic bit stream of $\mu_A(x)$. The output of comparator = 1 $\Leftrightarrow$ $\mu_A(x) > r$.

rules. The proposed architecture approximates weighted sum defuzzification. We restrict our analysis on triangular and trapezoidal membership functions, since those are commonly used in fuzzy control design. Moreover, singleton fuzzy sets are employed at the output of the FLC.

A. FUZZIFICATION

Fuzzification is the process of mapping the crisp measurements to the fuzzy interval $[0, 1]$. This fuzzy interval describes the membership of the fuzzy input variable. Let $\mu_A(x) \in [0, 1]$ denote the membership function of fuzzy set A, where the input space is X and $x \in X$ represents the crisp measurement. Our goal is to represent $\mu_A(x)$ in stochastic domain in order to simplify the inference mechanism, and thereby the whole architecture of FLCs.

The basic SC concept states that the stochastic representation is achieved by comparing the membership function value $\mu_A(x) \in [0, 1]$ to a uniform random number $r \in [0, 1]$ generated in every clock cycle, as it is shown in Fig. 2. The comparator provides the output pulses whenever the membership function value is larger than the generated random number. In this manner, the estimated value (or approximated value) $\hat{\mu}_A(x)$ is codified in the statistical mean value of the resulting pulse sequence.

Based on Fig. 2 a recognition follows. The procedure that the comparator produces 1's whenever the membership function value is bigger than the generated random number is no other than the procedure of producing 1's whenever the instantaneous crisp measurement x is in the strong α-cut set A_α of the fuzzy set A, where $\alpha \in [0, 1]$ is a uniform random number, the membership function μ_A represents the fuzzy

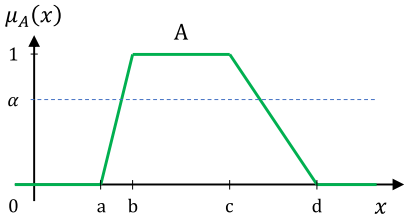


FIGURE 3. Trapezoidal membership function and its parameters.

set and $A_\alpha = \{x \mid \mu_A(x) > \alpha\}$. Therefore, the stochastic representation of the membership function value $\mu_A(x)$ is no other than its representation with series of α-cut set function values evaluated using the crisp x, where α randomly varies in every clock cycle in the range [0, 1]. Once the stochastic bit stream of $\mu_A(x)$ is generated, the estimation $\hat{\mu}_A(x)$ can be given based on Eq. (1) as:

$$\hat{\mu}_A(x) = \frac{1}{N}\sum_{i=1}^{N} \chi_{A_\alpha}(x)_i\,, \tag{9}$$

where $\chi_{A_\alpha}(x)$ denotes the characteristic function of the α-cut set of fuzzy set A, and N is the length of the stochastic sequence.

As a result of the aforementioned discussion, we can transform the problem of producing 1's based on the comparison of $\mu_A(x)$ and the uniform random number r to the problem of checking if the crisp measurement x is in the strong α-cut set that varies in every clock cycle. The analysis of checking if $\chi_{A_\alpha}(x)$ equals to 1 strictly depends on the chosen type of membership function. Let us recall the conditions for trapezoidal membership functions as it was described in [51]. The trapezoidal membership function is illustrated in Fig. 3 and its definition is given as follows.

$$\mu_A(x;a,b,c,d) = \begin{cases} 0, & x \le a \\ \dfrac{x-a}{b-a}, & a \le x \le b \\ 1, & b \le x \le c \\ \dfrac{d-x}{d-c}, & c \le x \le d \\ 0, & d \le x \end{cases} \tag{10}$$

It is straightforward to show, that the crisp measurement x belongs to the instantaneous α-cut set if

$$a + \alpha(b-a) < x \quad \wedge \quad x < d + \alpha(c-d)\,, \tag{11}$$

or its equivalent form if

$$\frac{b+a}{2} < x - \alpha(b-a) + \frac{b-a}{2} \quad \wedge \quad \frac{d+c}{2} > x + \alpha(d-c) - \frac{d-c}{2}. \tag{12}$$

Based on Eq. (11), the codification scheme of the membership function value $\mu_A(x)$ consists of comparing the crisp measurement x to the generated random number α that is properly scaled. The problem introduced first in Fig. 2 has been transformed to the coding circuit shown in Fig. 4. Fig. 4 forms the stochastic fuzzifier, whose output is the stochastic bit stream representing $\mu_A(x)$.

The architecture of stochastic fuzzifier simplifies further if symmetrical and triangular membership functions define the FLC as it will be shown in section VI. Applying the circuit illustrated in Fig. 4 the stochastic fuzzification is performed. Once the stochastic representations of membership function values are available the defined rules can be evaluated in the inference machine.

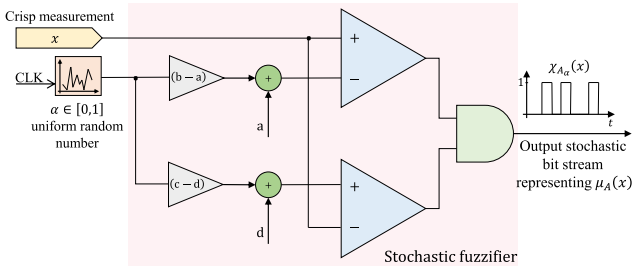


FIGURE 4. The stochastic fuzzifier. It generates the stochastic representation of $\mu_A(x)$ based on the crisp measurement x and the randomly varying α.

B. INFERENCE MACHINE AND AGGREGATION

The inference mechanism consists of assigning the so-called firing level to the output fuzzy set defined in each rule. The fuzzy operator (i.e., t-norm or t-conorm) is applied if the antecedent of a rule has more than one part. The output of the fuzzy operator is the firing level of the rule. This firing level represents the result of the antecedent evaluation. The antecedent can be combined with AND and OR methods. The most common arithmetic functions in the inference machine are the *minimum* and *product* for the AND method, and the *maximum* and *probabilistic or* for the OR method. These arithmetic functions are used in the evaluation of fuzzy rules.

Keeping in mind that the membership values are represented with stochastic bit streams, the aforementioned arithmetic functions can be effectively performed with the logic gates introduced in section II. Table 1 summarizes both these logic gates and their functionalities. Moreover, Fig. 5 depicts the coding schemes for executing the common arithmetic functions in stochastic domain.

TABLE 1. Summary of the stochastic arithmetic elements.

Gate	Functionality	Notes
AND	product	The input stochastic sequences shall be independent.
OR	probabilistic or	
AND	minimum	The input stochastic sequences shall be generated by the same random number generator.
OR	maximum	

The aggregation process combines the output fuzzy sets into a single fuzzy set. This paper focuses on singleton outputs, which results in a standard zero-order Sugeno system. The control design may require the inference mechanism

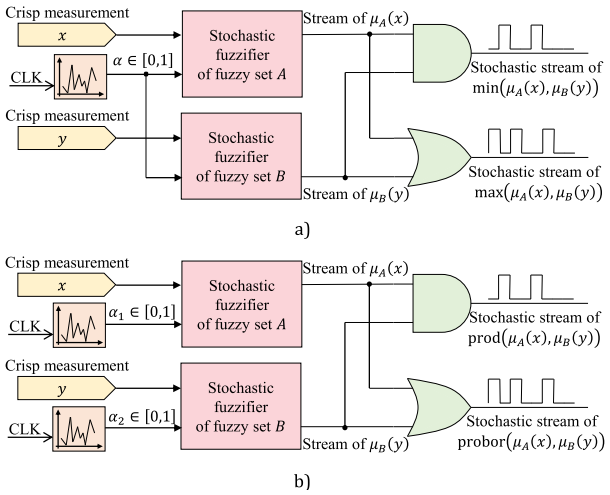


FIGURE 5. **Coding schemes for the stochastic logic-based fuzzy operators. (a) Min-max functions are generated with shared random number generator. (b) Product-probabilistic or functions are generated with independent random number generators.**

to aggregate the fuzzy outputs prior the calculation of the crisp output. In this case, the process consists of selecting the biggest firing level for each singleton output. As a result, the aggregation (i.e., finding the maximum) is performed by OR gates.

C. DEFUZZIFICATION

The defuzzification process maps the output fuzzy set back to crisp domain. Since the output fuzzy set is encoded with stochastic bit streams, therefore the defuzzification can be performed by a counter whose inputs are driven by the stochastic signals. In this manner, the crisp output is updated with a period of N clock cycles. On the other hand, the crisp output can also be obtained with LPFs. The LPF estimates the DC component of the stochastic stream. See the detailed description in section II. Exact techniques and solutions are described in section VI.

IV. THE SBR EMPLOYED IN THE EXPERIMENTS

A custom made SBR has been employed in the experimental validation. SBRs are the descendants of pendulum-cart systems and are characterized by advantageous electromechanical properties. In both education and research, these mechatronic systems provide wide application spectrum and are considered as important benchmark tools to verify novel control approaches [61]. The electromechanical properties, modeling and control solutions, and applications have been summarized in detail in recent works [61], [62]. Moreover, both the employed SBR system and its conventional fuzzy control approach have been presented in detail in our earlier works [63]–[65]; therefore, only the key information is described in the following paragraphs.

A. MECHATRONIC SYSTEM

Fig. 6 shows a photograph of the SBR. The hardware construction is built around two 16-bit ultra-low-power Texas Instruments MSP430F2618 microcontrollers (hereinafter MCU1 and MCU2). Low cost micro-electro-mechanical (MEMS) accelerometer and gyroscope sensors measure the IB dynamics, and additionally current sensors and two-channel incremental encoders are attached to both DC motors. The embedded electronic configuration has been described in detail in [63]–[65]. MCU2 is programmed as an inertial measurement unit (IMU). It collects the measurements from MEMS sensors and executes the Kalman filter to obtain the IB angle [66]. MCU1 executes the control tasks, i.e., it collects the measurements and drives the motors based on the applied control algorithm. Additionally, it continuously sends the instantaneous measurements to the PC via a wireless communication module.

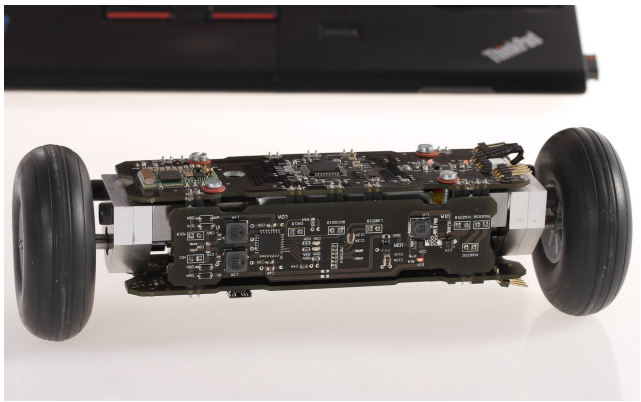


FIGURE 6. **Photograph of the employed mechatronic system (SBR).**

B. MATHEMATICAL MODEL

The system dynamics $\dot{x} = h(x, u)$ is described with an 8-dimensional nonlinear state space model as [61]:

$$\dot{x}(t) = \begin{bmatrix} \dot{q} \\ M(q)^{-1}\left(\tau_a - \tau_f - V(q,\dot{q})\right) \\ \frac{1}{L}\left(u - k_E k \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \end{bmatrix} \dot{q} - RI\right) \end{bmatrix},$$
$$y(t) = Cx(t), \tag{13}$$

where $x_{8\times 1} = (q, \dot{q}, I)^T$ denotes the state vector, $q = (\theta_1, \theta_2, \theta_3)^T$ is the vector of generalized coordinates. Moreover, $M(q)$ is the 3-by 3 inertia matrix and $V(q, \dot{q})$ is the 3-dimensional vector term including the Coriolis, centrifugal and potential force terms. The exact elements of these matrices have been derived in [63]. The output matrix C of the state space equation maps the state values to the output vector $y = (s, v, \theta_3, \omega_3)^T$, where $s = r(\theta_1 + \theta_2)/2$ is the linear displacement, $v = \dot{s}$ and $\omega_3 = \dot{\theta}_3$. The main parameters of the robot are summarized in Table 2.

TABLE 2. Notation of robot parameters.

Symbol	Unit	Value	Parameter name
θ_1, θ_2	rad	-	angular position of the wheels
θ_3	rad	-	angular position of the IB
I	A	-	vector of motor currents I_1, I_2
u	V	-	vector of motor voltages u_1, u_2
τ_a	Nm	-	vector of torques applied to the wheels
τ_f	Nm	-	vector of friction torques
l	mm	8.36	location of center of mass
r	mm	31.5	radius of the wheels
m_w	g	31.5	mass of the wheels
d	mm	177	distance between the wheels
m_b	g	360.4	mass of the IB
$J_{\mathcal{A}}$	gmm^2	81367	IB moment of inertia about $\mathcal{A}$ axis
$J_{\mathcal{B}}$	gmm^2	574620	IB moment of inertia about $\mathcal{B}$ axis
R	Ω	2.3	rotor resistance
L	μH	26	rotor inductance
k_E	mVs	2.05	back-EMF constant
k_M	mNm/A	2.05	torque constant
J_r	gmm^2	12	rotor inertia
f_m	mNms	0.021	viscous friction coefficient at the motors
f_w	mNms	0.18	viscous friction coefficient at the wheels
k	-	64	gear ratio of the gearbox

V. CONVENTIONAL FUZZY CONTROL SCHEME

First, the conventional fuzzy control (i.e., the reference control method with standard FLC architectures) is designed for the stabilization of the plant. This reference control scheme forms the basis for both the introduction of the equivalent SFLC-based control approach (discussed in section VI) and comparative analysis of the conventional solution and novel SFLC-based approach.

A. CONTROL STRATEGY

The control scheme realizes a closed-loop system, therefore it contains the plant (i.e., the dynamical system to be controlled) and the FLCs that provide the stabilizing control actions based on the feedback signals. The control scheme is required to simultaneously ensure the linear position of the robot and stabilize the inverted pendulum around the unstable upright position. An effective realization of this control task involves the application of three independent FLCs (i.e., FLC1 for linear position control, FLC2 for IB stabilization and FLC3 for yaw angle-based orientation control). This control scheme is described in detail in our earlier work [61]. Since the yaw angle control does not influence the relationship between the linear position and resulting IB oscillations significantly, therefore this paper omits the control of yaw angle of the robot and focuses on the design and implementation of FLC1 and FLC2.

Fig. 7 depicts the MATLAB/Simulink implementation of the employed control scheme. Since the controllers are implemented in digital domain, therefore both the data acquisition and control action calculation are executed at fixed $f_s = 1/T_s$ sampling rate. The cascade-connected scheme contains two PD-type FLCs, where FLC1 is responsible for the position control of the robot and FLC2 performs the stabilization of the inverted pendulum (IB). Both the membership functions and rule bases of these controllers have been introduced in [61]. The inputs of FLC1 are the linear position error e_s and its time derivative e_v, while the output of the controller

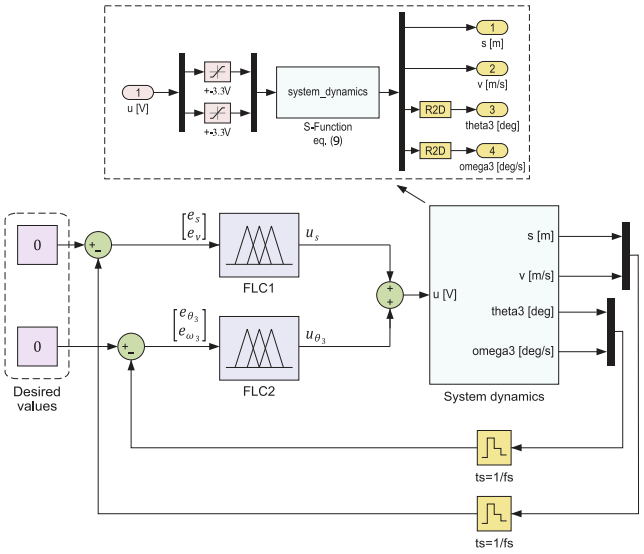


FIGURE 7. The applied fuzzy control scheme implemented in MATLAB/Simulink environment.

TABLE 3. The properties of the employed FLCs.

AND method	OR method	Output fuzzy set	Defuzzification method
Minimum	Maximum	Singletons	Weighted sum

is the control action u_s. Similarly, the inputs of FLC2 are the pendulum angle error e_{θ_3} and its time derivative e_{ω_3}, while the output of the controller is denoted with u_{θ_3}. The sum of these outputs constitutes the stabilizing control action and is supplied to both motors, i.e., the control action in the ith epoch is $u_i = u_{s,i} + u_{\theta_3,i}$, where the subscript i refers to the discrete time domain equivalent of the signals, e.g., $u_i = u(iT_s)$.

The design of these controllers have been described in detail in [61], therefore only the important results are discussed as follows. Table 3 summarizes the properties of the employed FLCs. Three membership functions characterize the input ranges for each FLC; these functions are uniformly distributed across the universes of discourse as shown in Fig. 8. The fuzzy sets N (negative), P (positive) and Z (zero) are used to define 9 fuzzy rules for each controller. The effective universes of discourse for the errors are defined with the following constants: $\theta_{3m} = 10$ deg, $\omega_{3m} = 150$ deg/s, $s_m = 0.2$ m and $v_m = 0.12$ m/s, while the output singleton values are defined with $u_{P,\theta} = u_{P,S} = 1$ V and $u_{N,\theta} = u_{N,S} = -1$ V values (see Fig. 8). The inference table (rule base) for these PD-like FLCs is defined in Table 4. Based on the controller properties indicated in Tables 3 and 4 and Fig. 8, the crisp control action applied to the motors in the ith epoch is given as:

$$u_i = \sum_{k=1}^{9} \kappa_{FLC1}^k \cdot \min\left(\gamma^k\left(e_{s,i}\right), \gamma^k\left(e_{v,i}\right)\right) + \sum_{k=1}^{9} \kappa_{FLC2}^k \cdot \min\left(\gamma^k\left(e_{\theta_3,i}\right), \gamma^k\left(e_{\omega_3,i}\right)\right), \quad (14)$$

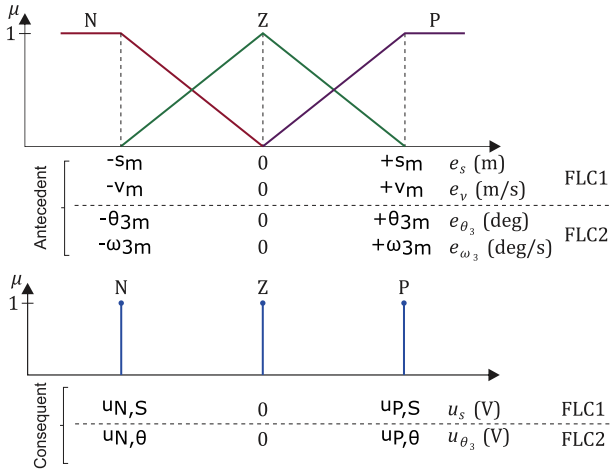


FIGURE 8. Membership functions of the IO variables of FLC1 and FLC2.

TABLE 4. Rule base of the employed FLCs.

Consequent $\{u_s, u_{\theta_3}\}$		Antecedent$_2$ $\{e_\nu, e_{\omega_3}\}$		
		N	**Z**	**P**
Antecedent$_1$ $\{e_s, e_{\theta_3}\}$	**N**	P	P	Z
	Z	P	Z	N
	P	Z	N	N

where $\gamma^k(\cdot)$ denotes the kth-rule fired membership function value for the corresponding error signal ($e_{s,i}$, $e_{v,i}$ for FLC1 and $e_{\theta_3,i}$, $e_{\omega_3,i}$ for FLC2) and κ^k represents the singleton value of the consequent of the kth rule (see Table 4 and Fig. 8). Based on the expert knowledge, the heuristic inference system may require to execute the aggregation of output fuzzy sets. In this case, the implication process outputs, truncated output functions $\kappa^k \cdot \gamma^k$ for $k = 1, \ldots, 9$, are combined into a single fuzzy set. The maximum, probabilistic or and sum operations are among the methods that are commonly used for aggregation.

B. CONTROL ANALYSIS

The characteristics of the control strategy introduced in the previous subsection is obtained using linear control synthesis tools. It is important to carry out this analysis, since the approximate stability margins determine the parameters of the stochastic defuzzifier, which is elaborated in subsection VI-C. Based on the heuristically defined fuzzy control strategy (see Fig. 7), the aim is to approximately obtain the lower bound of acceptable sampling frequency and the upper bound of acceptable time delay in the closed loop system. These values directly influence the accuracy of stochastic defuzzifier and thereby the accuracy of the estimation of crisp output from stochastic streams.

First the linear approximation of the control scheme is obtained. The nonlinear state-space representation (Eq. (13) with $u_1 = u_2 = u$) is linearized by evaluating the Jacobian matrix of system around the equilibrium point $(x_e, u_e) = (0, 0)$, i.e., $\dot{x} = h(x_e, u_e) = 0$. The system dynamics can be written with its Taylor expansion [67]:

$$\dot{x} = \left(\frac{\partial h}{\partial x}\right)_{(x_e=0,u_e=0)} x + \left(\frac{\partial h}{\partial u}\right)_{(x_e=0,u_e=0)} u + h_{h.o.t}(x, u), \tag{15}$$

where $h_{h.o.t}$ indicates the higher-order terms in x and u. Denoting with A the Jacobian of h with respect to x at the equilibrium point, and with B the Jacobian of h with respect to u at the same point, Eq. (16) describes both the linearization of the nonlinear system at the equilibrium $(x_e, u_e) = (0, 0)$ and the system transfer matrix $G(p)$ for the output vector $y = C'x = (s, \theta_3)^T$ [68]:

$$\begin{aligned} \dot{x} &= Ax + Bu, \\ G(p) &= C'(sI - A)^{-1} B + D. \end{aligned} \tag{16}$$

Let the transfer $G_1(p)$ denote the input-output relationship between the linear position $S(p)$ and control input $U(p)$ in frequency domain. Similarly, let the transfer $G_2(p)$ denote the relationship between the pendulum angle $\Theta_3(p)$ and control signal $U(p)$, where p is the Laplace operator, i.e.:

$$G_1(p) = \frac{S(p)}{U(p)}, \quad G_2(p) = \frac{\Theta_3(p)}{U(p)}. \tag{17}$$

In order to obtain the open- and closed-loop transfer function, the fuzzy PD controllers (FLC1 and FLC2 introduced in section V) are converted to nonfuzzy PD controllers. If $[-e_{max}, e_{max}]$, $[-\dot{e}_{max}, \dot{e}_{max}]$, and $[-u_{max}, u_{max}]$ denote the effective universes of discourse for the errors e, $\dot{e}$, then the fuzzy PD controller is equivalent to the following nonfuzzy PD controller [69]:

$$u_{PD}(t) = \frac{u_{max}}{e_{max}} e(t) + \frac{u_{max}}{\dot{e}_{max}} \dot{e}(t). \tag{18}$$

Denoting the transfer functions of FLC1 and FLC2 with $C_1(p)$ and $C_2(p)$, respectively, and using the membership function parameters $\{s_m, \nu_m, \theta_{3m}, \omega_{3m}\}$ illustrated in Fig. 8, $C_1(p)$ and $C_2(p)$ are obtained by expanding Eq. (18) to FLC1 and FLC2. Namely:

$$\begin{aligned} u_{C_1}(t) &= -\frac{e_s(t)}{s_m} - \frac{\dot{e}_s(t)}{\nu_m}, \text{ thus} \\ C_1(p) &= -\frac{1}{s_m} - \frac{1}{\nu_m} \frac{p}{\tau p + 1}, \end{aligned} \tag{19}$$

$$\begin{aligned} u_{C_2}(t) &= -\frac{e_{\theta_3}(t)}{\theta_{3m}} - \frac{\dot{e}_{\theta_3}(t)}{\omega_{3m}}, \text{ thus} \\ C_2(p) &= -\frac{1}{\theta_{3m}} - \frac{1}{\omega_{3m}} \frac{p}{\tau p + 1}, \end{aligned} \tag{20}$$

where the pseudo-derivative was introduced with the filter constant τ. Once the transfer functions $G_1(p)$, $G_2(p)$, $C_1(p)$ and $C_2(p)$ of the control scheme illustrated in Fig. 7 are known, the open-loop and closed-loop transfer functions can be obtained. Let $W_{ol,S}(p)$ and $W_{cl,S}(p)$ denote the open loop and closed loop transfer functions related the output $S(p)$, then:

$$W_{ol,S} = C_1(p) W_{\Theta_3}(p) G_1(p), \tag{21}$$

$$W_{cl,S} = \frac{C_1(p)\, W_{\Theta_3}(p)\, G_1(p)}{1 + C_1(p)\, W_{\Theta_3}(p)\, G_1(p)}, \tag{22}$$

where $W_{\Theta_3}(p) = \frac{1}{1+C_2(p)G_2(p)}$ denotes the inner dynamics related to $\Theta_3(p)$.

The phase margin (PM) is inspected first, since this value describes the damping of the system, moreover, it gives the bound for adequate stabilization. PM describes the acceptable extra phase lag in the closed-loop system before it becomes unstable. The Bode diagram of $W_{ol,S}$ shows that the system is characterized by $PM \approx 52.7\ deg$ and $\omega_c \approx 2.33\ rad/s$ crossover frequency. Based on this result, the acceptable time delay can be deduced in the system. The delay $G_D(p) = e^{-pT_d}$ does not influence the magnitude (i.e., $|G_D(p)| = 1$ for all ω frequencies, where $p = j\omega$), but it shifts the control signal in time, where the phase is $\angle G_D(j\omega) = -\omega T_d$. This enables the calculation of the bound of stability in terms of time delay. Namely, the ratio of PM and crossover frequency results in $PM/\omega_c = 0.395\ s$ delay margin in our problem. Let $PM_{min} = 40\ deg$ be the lowest adequate PM value, which is a general rule of thumb in control engineering problems, then the acceptable bound of time delay T_d is obtained as $T_{d1,max} = (PM - PM_{min})/\omega_c \approx 0.1\ s$ for FLC1. This result is verified in Fig. 9. Based on similar findings, the acceptable time delay for FLC2 is obtained as $T_{d2,max} = 0.1\ s$. These results give crude approximations for the acceptable time delays in the closed-loop system. Fig. 10 both highlights the effect of the delay on the system dynamics and verifies the aforementioned estimation process. It can be observed that 0.2 s time delay is the bound of stabilization, i.e., this delay margin results in oscillatory behavior. Since the feasible LPF effect was inspected in the aforementioned investigation, therefore the crude approximation of $e^{-T_d s} \approx \frac{1}{1+T_d s}$ was utilized in the step response simulation in Fig. 10.

Next, the sampling frequency f_s is analyzed. Note that the sampling frequency equals to the control frequency, i.e., the rate at which measurements are sampled is the same as the rate at which control actions are supplied to the plant. In general, too fast sampling results in decrease in accuracy [70]. This fact is especially true for the proposed stochastic architecture, since the amount of data used in stochastic defuzzifier is inversely proportional to the sampling frequency (see the details in subsection VI-C). Therefore, the lower bound of sampling frequency $f_{s,min}$ is inspected, which maximizes the stochastic bit stream length in the defuzzification process and thereby determines the feasible accuracy of the crisp output estimate. As it was already discussed at the end of section II, this represents a trade off between accuracy and system dynamics, since the lower the control frequency the more accurate control signal estimate is generated, however the inherent time delay poses constraint on the achievable bandwidth.

The lower limit for acceptable sample rate which satisfies all performance specifications is obtained as follows. Both the reference tracking performance and effect of prefiltration determines f_s. It should be also considered that a Kalman

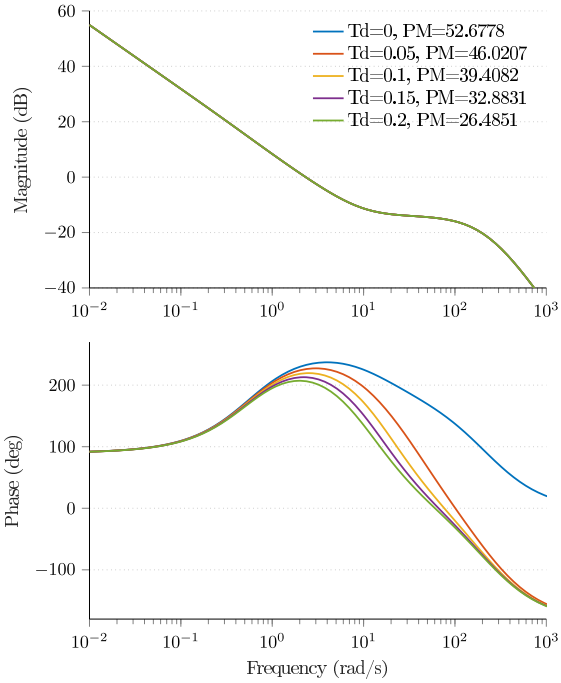


FIGURE 9. Bode diagram of $W_{ol,S}$ for different time delays.

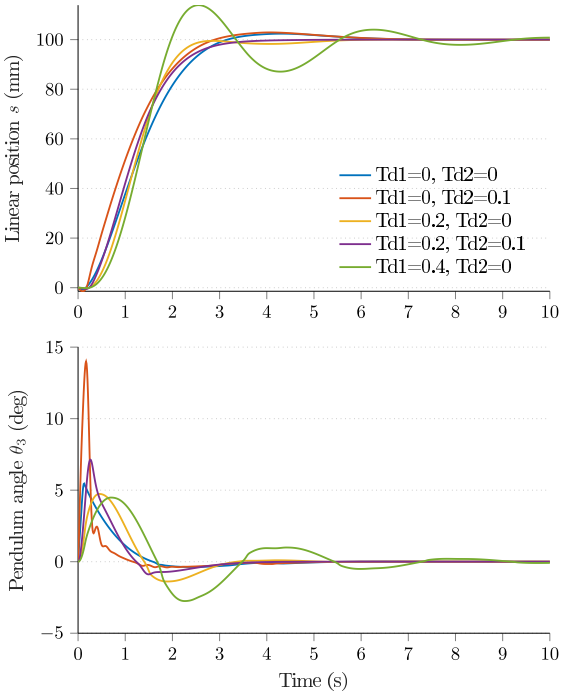


FIGURE 10. Step response of the closed-loop system for different time delays.

filter estimates the pendulum orientation in the closed loop system. To both cancel out system instabilities and provide the desired performance the sample rate should be bigger than $100 \times \omega_{BW}$, which is a common choice in these control engineering problems [70]. The bandwidth of the closed loop

system $W_{cl,S}$ is $\omega_{BW} = 4.14\ rad/s$, therefore the theoretical lower bound for sampling rate is $f_{s,min} = 65\ Hz$.

VI. DESIGN OF THE SL-BASED FLCs

This section introduces the equivalent stochastic-based fuzzy architecture of the control scheme introduced in the previous section. During the design procedure the same membership functions, ranges, rules, input-outputs and inference properties are employed (see Fig. 8 and Tables 3 and 4), moreover, the same control scheme is realized as the one introduced in Fig. 7. However, instead of the conventional FLCs, the stochastic fuzzy elements described in section III are applied to realize the equivalent SFLCs. Since FLC1 and FLC2 are characterized by identical structures, we present the design steps of the equivalent stochastic architecture for FLC1. Then, at the end of the section, the architecture is expanded to FLC2.

A. FUZZIFICATION

Fig. 8 shows that both inputs are classified into three triangular fuzzy sets, namely, P (positive), N (negative) and Z (zero) fuzzy sets characterize the input ranges. The fuzzy sets for the displacement error e_s are defined as $N = [-\infty, -s_m, 0]$, $Z = [-s_m, 0, +s_m]$ and $P = [0, +s_m, +\infty]$, while the fuzzy sets for the velocity error e_v are given as $N = [-\infty, -v_m, 0]$, $Z = [-v_m, 0, +v_m]$ and $P = [0, +v_m, +\infty]$. Since the AND method is executed with minimum operator (see Table 3), therefore the fuzzification procedure requires eight comparators (two for each zero fuzzy set, and one for each positive or negative fuzzy set), and a shared random number generator whose output is properly scaled.

Based on Eq. (11), the conditions to check if the inputs are in the α-cut set can be evaluated. As an example, the input $e_{s,i}$ in the ith epoch is in the α-cut set of the fuzzy set Z if and only if

$$\chi_{Z_\alpha}\left(e_{s,i}\right) = 1 \Leftrightarrow \left(e_{s,i} > s_m\alpha - s_m \wedge e_{s,i} < -s_m\alpha + s_m\right). \quad (23)$$

Similarly for P and N fuzzy sets, one obtains the conditions as:

$$\chi_{P_\alpha}\left(e_{s,i}\right) = 1 \quad \Leftrightarrow \quad e_{s,i} > s_m\alpha, \quad (24)$$

$$\chi_{N_\alpha}\left(e_{s,i}\right) = 1 \quad \Leftrightarrow \quad e_{s,i} < -s_m\alpha. \quad (25)$$

The conditions are similar for the second input e_v, i.e., Eqs. (23), (24) and (25) can be directly applied, only the corresponding constant v_m is used instead of s_m. The realization of the stochastic fuzzification scheme is straightforward based on the aforementioned conditions and is shown for both inputs (e_s and e_v) in Fig. 11. The outputs of the stochastic fuzzifiers are stochastic bit streams that represent the membership function values of each input for the fuzzy sets N, Z and P. Once the membership values are represented with randomly varying bit-streams the logic operators introduced in section III can be applied to perform the chosen inference mechanism.

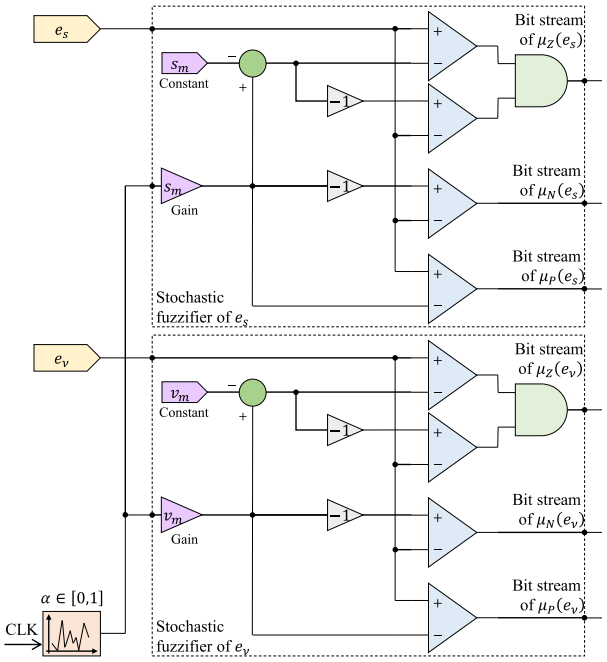


FIGURE 11. Fuzzification scheme for the inputs e_s and e_v.

B. INFERENCE MECHANISM AND AGGREGATION

In the reference FLC structure, the minimum logic operation is employed for the AND method (see Table 3). Moreover, Table 1 summarized that if the stochastic bit streams are generated with a shared random number generator, then these signals can easily be compared with simple logic gates. As a result, the minimum operation is performed with simple AND gates in the implication phase. Additionally, the aggregation of firing levels consists of selecting the biggest firing value for every singleton output. Therefore, the maximum operation is performed with a simple OR gate for every output fuzzy set.

Based on the rule base defined in Table 4, the scheme of the stochastic-based inference machine along with its aggregation phase are illustrated in Fig. 12. This architecture can be directly expanded for FLC2 using the inputs e_{θ_3} and e_{ω_3} and membership function parameters θ_{3m} and ω_{3m}.

C. DEFUZZIFICATION

In order to obtain the control action estimate, the stochastic representation has to be converted back to crisp domain. This process is the task of the defuzzification unit. The inputs of the defuzzification are the aggregated stochastic bit streams that represent the crisp control action. Let f_{inf} denote the inference frequency, i.e., the clock rate at which the stochastic bit streams are generated. Moreover, let f_{con} denote the control frequency, i.e., the rate at which control action is supplied to the plant. Essentially, $f_{con} = f_s$ in our application, however it is reasonable to use separate notations, since these variables can be different in stochastic control if the application requires.

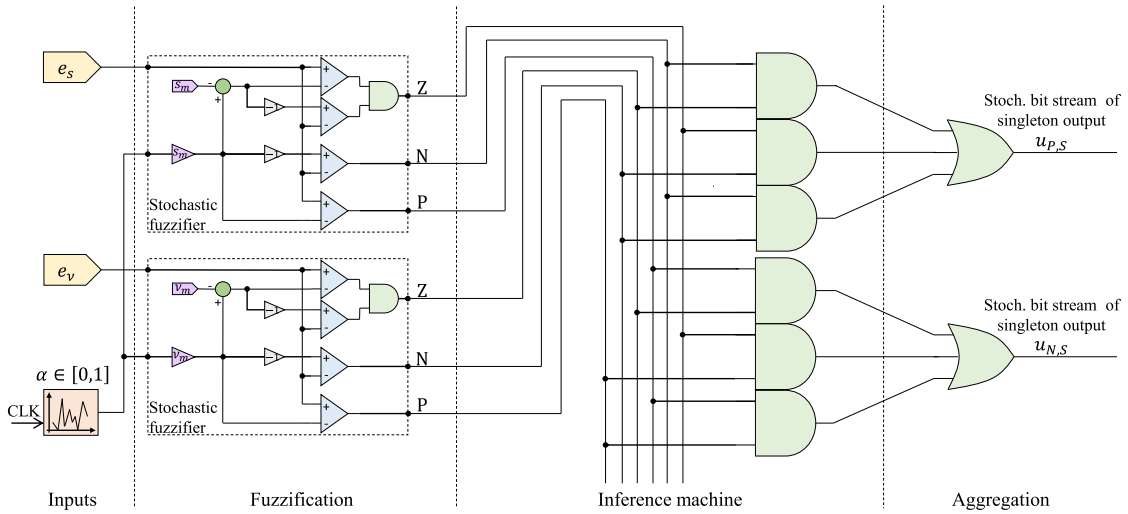


FIGURE 12. Architecture of the stochastic FLC (SFLC1): from crisp inputs, over fuzzification, to inference and aggregation.

The stochastic defuzzifier uses sequences of bit streams that represent the singleton outputs in stochastic domain at the input rate $f_{inf} = Nf_{con}$ (e.g., see $u_{P,S}$ and $u_{N,S}$ stochastic streams at the right side of Fig. 12). A decimation process should be executed, which estimates the crisp output signal at the lower f_{con} output rate. This decimation process performs both averaging and rate reduction of the input stochastic bit streams. As it was discussed in section II, the literature describes different methods for this decimation process, i.e., it can be realized by simple counters and LPFs. In this article both methods have been implemented and evaluated.

It is concluded based on Figs. 7, 8 and 12 that the aggregation process generates two parallel stochastic stream lines for each controller. These signals represent the singleton outputs in stochastic domain (i.e., $u_{P,S}$ and $u_{N,S}$ for FLC1, and $u_{P,\theta}$ and $u_{N,\theta}$ for FLC2). Therefore, the inputs of the stochastic defuzzifier are the SL-based singleton outputs at f_{inf} frequency; these signals are combined via simple logic gates, then the counter performs the accumulation. Namely, if $u_{P,S}$ is 1 and $u_{N,S}$ is 0, then the counter is incremented, else if $u_{P,S}$ is 0 and $u_{N,S}$ is 1, then the counter is decremented; in other cases the counter value is not changed. If the decimation process is performed solely with a counter, then the accumulated value is normalized by $N = f_{inf}/f_{con}$ and the counter value is reset to 0. Therefore, the size of the counter is defined by the decimation ratio $N = f_{inf}/f_{con}$, and the counter is reset at f_{con} rate. The actual value of the counter defines the associated crisp output (i.e., the voltage value to be applied for stabilization). The counter is less tunable to the dynamics of the system compared to a LFP. The forward Euler method-based transfer function is characterized by only the decimation ratio:

$$H(z) = \frac{f_{con}}{f_{inf}} \frac{1}{\left(1 - z^{-1}\right)}. \tag{26}$$

In case of LPF-based decimation process, first the counter is used to accumulate the input stochastic signals, i.e., it is reset at f_{inf} rate. Then, the LPF characterized by f_0 cut off frequency executes the information processing and provides the crisp output signal. The transfer function of a digital first order LPF is given as:

$$H(z) = \frac{1 + z^{-1}}{\frac{\omega_0 T_{inf} + 2}{\omega_0 T_{inf}} + \frac{\omega_0 T_{inf} - 2}{\omega_0 T_{inf}} z^{-1}}, \tag{27}$$

where $\omega_0 = 2\pi f_0$ and $T_{inf} = 1/f_{inf}$ indicates the inference period.

In both cases, the control signal estimate ($\hat{u}_s$ in case of SFLC1 and $\hat{u}_{\theta_3}$ in case of SFLC2) can be updated at f_{con} rate. However, the LPF-based method enables the f_{inf}-based update rate as well. The control signal estimate can be characterized by different error metrics, e.g., the mean error μ_e (ME), mean absolute error $\mu_{|e|}$ (MAE), and standard deviation σ_e (STD) describe the estimation performance. Moreover, the effective bit depth L of the approximation can be inspected for different f_{con}, f_{inf} and f_0 configurations. The structure of the defuzzification unit for SFLC1 is depicted in Fig. 13. The output of this unit is the approximation (or estimation) $\hat{u}_s$ of u_s, i.e., SFLC1 provides $\hat{u}_s$, while the reference FLC1 generates u_s. The same structure is employed for SFLC2, only the input stochastic signals are $u_{\theta_3,N}$ and $u_{\theta_3,P}$ in that case.

1) COUNTER-BASED DEFUZZIFICATION

It is easy to deduce, that the bigger the decimation ratio $N = f_{inf}/f_{con}$, the higher resolution (i.e., more precision) of the control signal $\hat{u}_s$ is obtained. If feasible inference frequency f_{inf} and proper control frequency f_{con} are selected, then the precision of the control signal can be determined. The acceptable minimal control frequency $f_{con,min}$ satisfies the Shannon Theorem, prevents signal distortion due to aliasing, moreover,

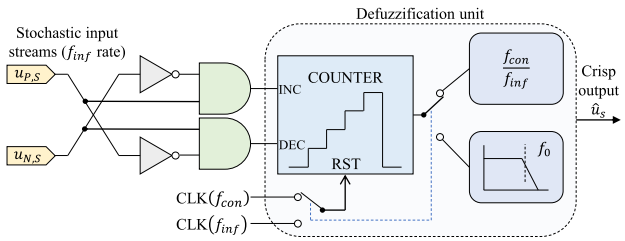


FIGURE 13. The stochastic defuzzification unit for SFLC1.

it keeps the PM in order to ensure stability. In subsection V-B it was obtained that the lower bound of the control frequency is $f_{con,min} = f_{s,min} = 65\ Hz$. It is interesting to inspect the achievable control signal precision for a particular f_{con}.

For the analysis, let the control frequency be $f_{con} = 100\,Hz$, which maintains an appropriate PM at the crossover frequency (see Fig. 9). Fig. 14 depicts the evaluation of the SFLC output for different inference frequencies, i.e., $f_{inf} = \{5, 50, 500\}\ kHz$ frequencies were selected for the analysis. This step response simulation well indicates that the approximation of the reference FLC (FLC1) becomes more and more precise as the decimation ratio increases. The first row of Fig. 14 shows the linear position and pendulum angle values during the stabilization process, respectively. The second row

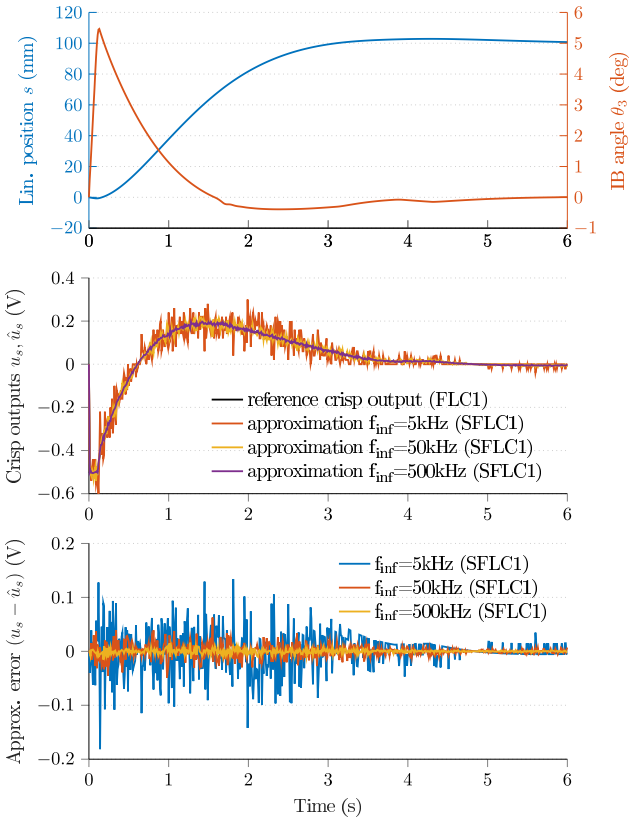


FIGURE 14. Crisp output estimate $\hat{u}_s$ for different inference frequencies during the step response of the fuzzy control scheme.

highlights the crisp output u_s generated by the reference FLC (see the black curve) and the SL-based approximations $\hat{u}_s$ for different inference frequencies (i.e., red curve shows the outcome for $N = f_{inf}/f_{con} = 50$, yellow curve corresponds to $N = 500$, while the purple curve is related to $N = 500$). Finally, the third row highlights the approximation error $e = u_s - \hat{u}_s$, i.e., the difference between the reference FLC output and stochastic approximation for different f_{inf} configurations.

It is also interesting to observe how the stochastic fuzzy surface gets distorted with different decimation ratios compared to the original fuzzy surface. The fuzzy surface of FLC1 was evaluated in Fig. 15. It can be observed that the SL-based approximation becomes more precise as the inference speed is increased. The top-left corner shows the reference fuzzy surface, where the crisp control action u_s is generated as a function of the inputs e_s and e_v. This smooth surface was evaluated in 40×40 points, i.e., the output was obtained by raking through the input ranges point-by-point. The top-right corner highlights the approximation of the reference control surface for decimation ratio $f_{inf}/f_{con} = 50$. It is seen that the SFLC well reconstructs the characteristics of the reference FLC, however notable approximation errors are present. The bottom-left plot highlights the approximation for $f_{inf}/f_{con} = 500$, which clearly provides improved approximation quality, i.e., significantly less imperfection characterizes the control surface. Finally, the bottom-right plot shows the approximated surface for decimation ratio $f_{inf}/f_{con} = 5000$, where basically no visible difference can be observed compared to the reference control surface (top left plot).

The approximation errors $e = u_s - \hat{u}_s$ are highlighted for different decimation ratios on histograms in Fig. 16, where the difference between the reference fuzzy surface and its SL-based approximation was evaluated on a 100×100 grid. This analysis also highlights that the inference speed directly determines the achievable precision. It can be observed in Fig. 16 that by increasing the decimation ratio the STD (σ_e) significantly decreases from 59 mV to 1.8 mV. Similarly, the range of error e is reduced from $\pm 200\ mV$ to $\pm 5\ mV$. The ME (μ_e) tends to converge to zero as the decimation ratio is increased; this is confirmed in each scenario. In addition, the MAE ($\mu_{|e|}$) also follows the same tendency, i.e., $\mu_{|e|} = 45\ mV$ in case of the crude approximation of $f_{inf}/f_{con} = 50$, while at a high inference frequency this value is decreased to 1.4 mV. The tendency of error metrics is depicted in Fig. 17.

The effective bit depth of the crisp output estimate was also inspected, see the results in appendix B. The mean value μ_L of the achieved bit depth L can be approximated as:

$$\mu_L \approx \frac{1}{2}\log_2\left(\frac{f_{inf}}{f_{con}}\right) + exponent, \tag{28}$$

where the *exponent* is 8 or 11 for single-precision or double-precision representation, respectively. This approximation is verified empirically in Fig. 27, where both the reference fuzzy controller (FLC1) and its stochastic approx-

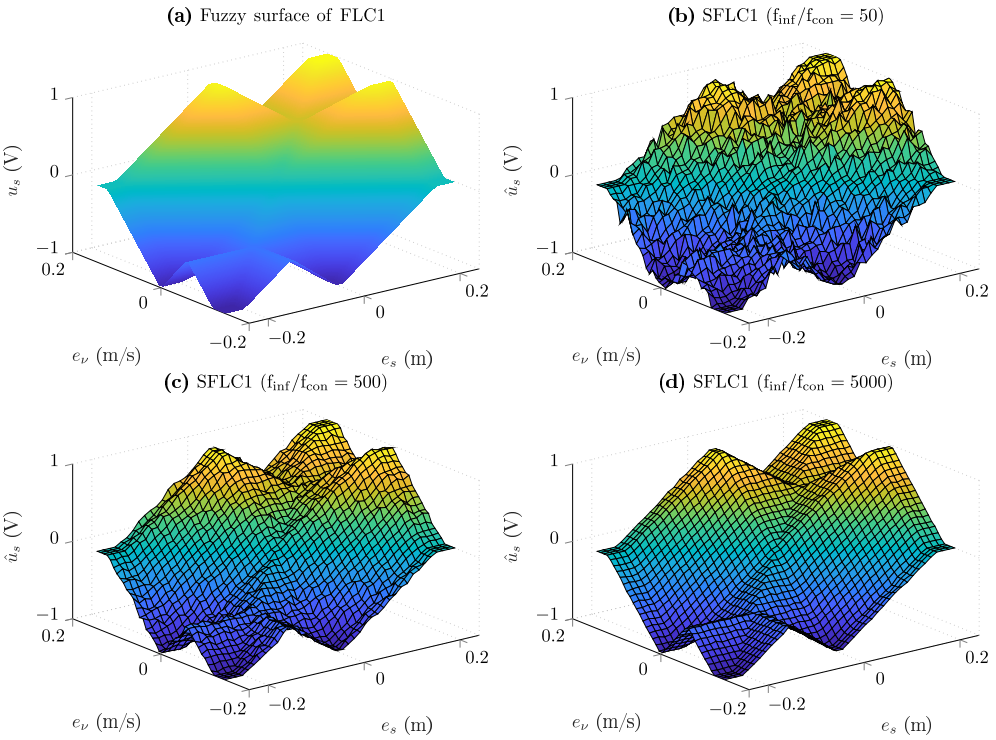


FIGURE 15. Fuzzy surface as a function of f_{inf}: (a) surface of reference controller (FLC1), (b) stochastic approximation of FLC1 at decimation ratio 50, (c) stochastic approximation of FLC1 at decimation ratio 500 and (d) stochastic approximation of FLC1 at decimation ratio 5000.

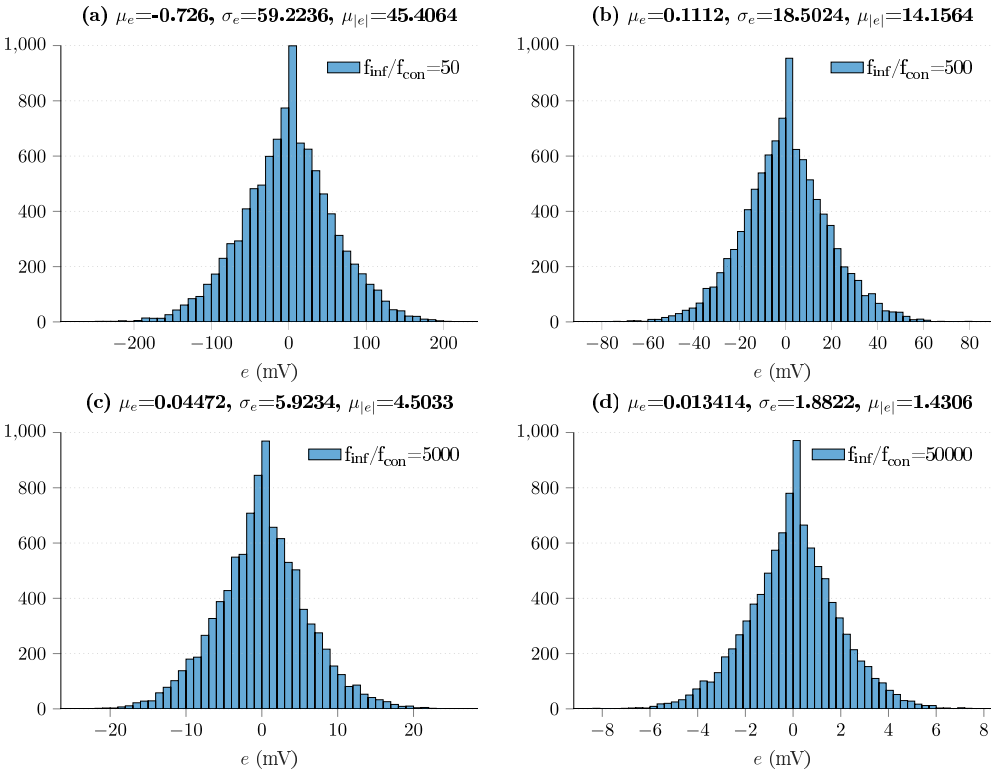


FIGURE 16. Error metrics for different decimation ratios (N): (a) $N = 50$, (b) $N = 500$, (c) $N = 5000$ and (d) $N = 50000$.

TABLE 5. The characteristics of the crisp output for different configurations.

f_{inf}/f_{con}	**50**	**100**	**500**	**50000**
f_{inf} (kHz)	5	10	50	5000
μ_e (mV)	0.116	0.036	0.0026	0.000055
$\mu_{\|e\|}$ (mV)	45.46	32.09	14.36	1.43
σ_e (mV)	59.40	41.97	18.79	1.88
μ_{L64} (bits)	13	14	15	18

imation (SFLC1) were evaluated in 10^6 different input combinations.

Based on both the aforementioned analysis and results depicted in Figs. 17 and 27, it is concluded that the defuzzification process should satisfy two contrary requirements. On one hand, the higher the control frequency f_{con} the less stability decrease is realized (i.e., PM is not decreased during the decimation process significantly), however this also results in decrease in control signal precision. On the other hand, the motivation for lowering the inference rate f_{inf} is cost, since the digital hardware should be capable of processing the calculations. However, the lower inference speed f_{inf} also results in decrease in control signal precision. The choice of f_{inf} and f_{con} depends on the microprocessor, peripherals used for digital analog conversion and the driving circuits employed in the embedded system. In our case, a general purpose timer peripheral is used to generate the PWM signals, which drive the H-bridge motor driver circuit. Therefore, the resolution of the PWM generator both poses another constraint and sets also an upper bound for the acceptable approximation error. Some notable numbers that satisfy the aforementioned contrary requirements are summarized in Table 5, where the control frequency was fixed at $f_{con} = 100$ Hz.

2) LPF-BASED DEFUZZIFICATION

If LPF performs the defuzzification, then it shall have small impact on the PM. The LPF supplies the average of the stochastic input stream via integration, moreover, it also reduces the modulation of high-frequency noise to lower frequency components (aliasing). The dynamics of the filter is determined by two parameters based on Eq. (27), i.e., the inference and cut-off frequencies (f_{inf} and f_0) determine both the smoothness and precision of output control signal. Similarly to the counter-based defuzzification method, the aforementioned two parameters pose constraints on the design process. Since the DC component of the stochastic stream represents the approximation of the reference crisp output, therefore f_{inf} should be increased to enable the integration of many stochastic pulses. In addition, the accurate estimation requires f_0 to be decreased, i.e., the LPF should be characterized by slow dynamics. The cut-off frequency f_0 determines the filter dynamics, whose lower bound should be analyzed from control systems standpoint, since the lower the cut-off frequency the more stability and performance issue may present in the closed loop system.

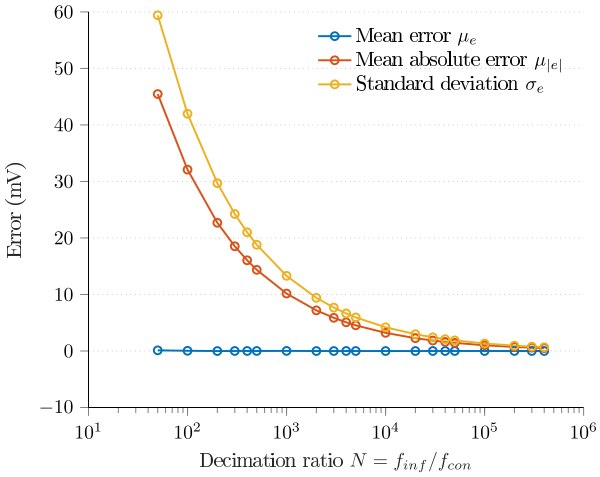


FIGURE 17. Error characteristics for different decimation ratios (*N*).

From control systems perspective, the cut-off frequency of the filter should be significantly higher than the system bandwidth, thereby the filter dynamics does not decrease the closed loop performance quality (i.e., the phase lag will not be detrimental to the system stability). It was obtained in subsection V-B that the maximum time delay introduced by the filter stage is $T_{d,max} = 0.1\ s$, which decreases the PM to 40 *deg*. At $T_{d,max} = 0.1\ s$ the corresponding filter cut-off frequency can be calculated as $f_0 = 1/\left(2\pi T_{d,max}\right) \approx 1.6\ Hz$. However, it should be also noted that this filter configuration is characterized by $\tau_s = 4\ T_{d,max} = 0.4\ s$ rise time. An $f_0 = 6\ Hz$ cut-off frequency is a more reasonable choice, since the dynamic response of the estimation process is faster. Moreover, the influence of the filter can be ignored in the closed-loop, since breakpoint will be located at 10 times the system bandwidth ω_{BW}. This reasoning is verified clearly by the results of subsection V-B. Namely, the minimum sampling rate was obtained to be $f_{s,min} = 65\ Hz$, which is approximately 10 times higher than $f_0 = 6\ Hz$, therefore appropriate reduction in the high-frequency noise is achieved at $f_{s,min}/2$. The higher cut-off frequency f_0 requires higher inference frequency f_{inf} as well, since the output ripple magnitude is directly influenced by the ratio f_{inf}/f_0.

Fig. 18 depicts the performance of LPF-based defuzzification, where the LPF cut-off frequency was set to $f_0 = 6\ Hz$ and $f_{inf} = \{5, 50, 500\}\ kHz$ inference frequencies were utilized in the SFLC architecture. The first row shows the step response for $s_d = 0.1\ m$, where the blue curve corresponds to the realized linear position s and the red curve highlights the pendulum angle θ_3 during the stabilization process. The second row of Fig. 18 shows the crisp output signals of the controllers (both for FLC1 and SFLC1). Namely, the black curve corresponds to the original FLC (i.e., the reference crisp output u_s), the other three curves highlight the approximation $\hat{u}_s$ for different f_{inf}/f_0 defuzzification configurations. Finally, the third row depicts the approximation errors ($e = u_s - \hat{u}_s$). It can be concluded that the $f_0 = 6\ Hz$ cut-off frequency

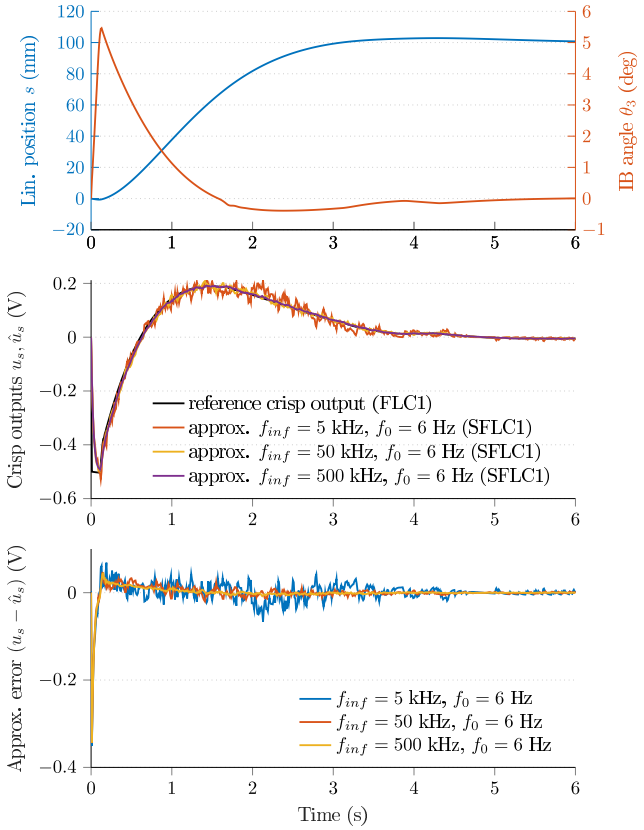


FIGURE 18. Crisp output estimate $\hat{u}_s$ for different f_{inf}/f_0 configurations during the step response of the fuzzy control scheme.

introduces an acceptable time delay in the system. Moreover, the LPF-based defuzzification stage provides the approximation of reference crisp control signal to satisfactory degree. As it was expected, the inference speed f_{inf} determines the deviation of the approximated crisp outputs $\hat{u}_s$, i.e., the higher the inference frequency the less ripple magnitude characterizes the filter output. The characteristics of the approximation error is left open to be analyzed in our future study.

VII. SIMULATION RESULTS

The simulation of the proposed control schemes was performed in MATLAB/Simulink environment. The nonlinear state space representation of the plant (Eq. (13)) was implemented using an S-Function block, while the reference FLCs were designed with the Fuzzy Logic Toolbox of MATLAB. Then, the equivalent stochastic-based controllers (SFLCs) were designed and implemented in MATLAB function blocks based on the reference fuzzy parameters (see Eqs. (23), (24), (25), Fig. 8 and Table 3). Both the LPF and counter-based defuzzification methods were evaluated in the simulation environment. The control frequency was set to $f_{con} = 100$ Hz; $f_{inf} = \{5, 50\}$ kHz inferences frequencies were evaluated during the stabilization of the plant. The LPF was characterized by $f_0 = \{1.6, 6\}$ Hz cut-off frequencies which both prevent the aliasing effects and provide limited PM decrease. Regarding the counter-based decimation, the ratio f_{inf}/f_{con} determined the length of the counter; the output signal was obtained by normalization (see Fig. 13). The dynamics of the robot was sampled at fixed $f_s = 100$ Hz sampling frequency. The same membership functions, ranges, and control scheme were applied for the both fuzzy architectures. Hence, the influence of the stochastic approximation on the closed-loop performance can be observed in Figs. 19 and 20.

The simulation results depict the step response of the closed-loop system for both the reference fuzzy control approach and its SFLC-based approximations in Figs. 19 and 20. In these figures, the first row shows the reference tracking dynamics for the linear position of the robot (the black curve is the desired position s_d, the blue curve corresponds to the robot position s realized with the reference fuzzy scheme, while the red, yellow and green curves correspond to the robot position achieved by the stochastic control circuits). The second row depicts the robot velocity v during the stabilization process. The third row highlights the resulting pendulum angle θ_3, while the fourth row shows the angular velocity of the IB (ω_3). Finally, the last row depicts the crisp output $u = u_s + u_{\theta_3}$ (i.e., the applied voltage to the DC motors) for the reference fuzzy control and its approximations. The simulation results show that the applied fuzzy control schemes (both the reference controllers and their stochastic approximations) successfully stabilize the system, fulfill the control requirements and provide an acceptable reference tracking performance.

In case of counter-based defuzzification (see Fig. 19), the decimation ratio $N = f_{inf}/f_{con} = 50$ resulted in a slightly poorer control performance (red curve). It can be observed that this configuration stabilized the system successfully, however notable oscillations were produced at around 4 s, after the pendulum was returned to the vertical upright position. This outcome was expected, since the resolution of the control signal (i.e., the crisp output of SFLC) was relatively small. Namely, the smallest increment in the crisp output signal was 0.02 V, which kept the pendulum in the vicinity of the upright position. A much higher performance was achieved with $N = f_{inf}/f_{con} = 500$ decimation ratio (see the yellow curve), since this configuration enabled to approximate the crisp output with higher resolution. In this case the smallest increment was 0.002 V in the control signal, and it can be seen that there is no notable difference between the performance of reference FLC and SFLC-based control schemes. Basically, the provided output signal resolution enabled to provide both the same system conditions and control performance as the reference control approach.

The stochastic scheme, which employed LPF-based defuzzification in the SFLCs, also produced satisfying control performances (see Fig. 20). This approach slightly decreased the PM of the closed-loop system, however this decrease did not have notable impact on the reference tracking performance. It can be seen in the first row, that each configuration (i.e., both the 1.6 Hz, and 6 Hz cut-off

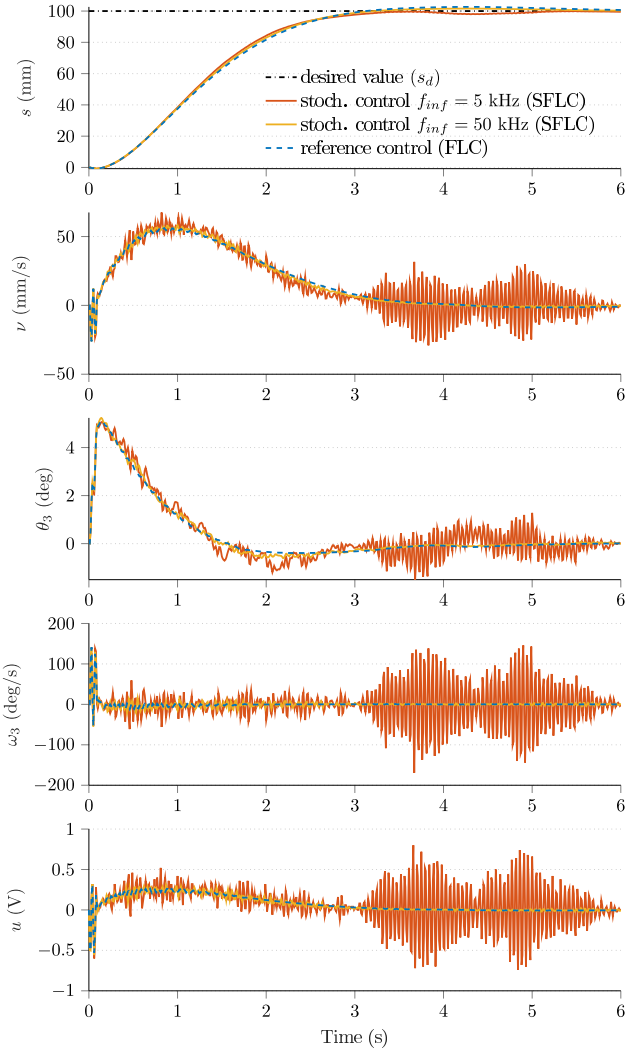


FIGURE 19. Simulation results: stabilization of the SBR with the stochastic control scheme employing counter-based defuzzification.

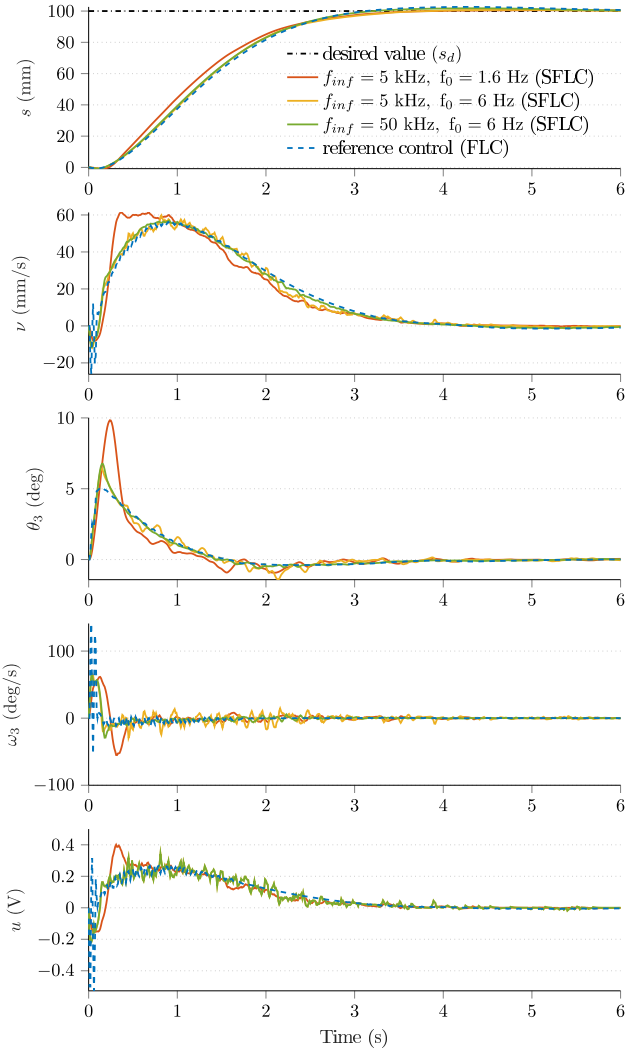


FIGURE 20. Simulation results: stabilization of the SBR with the stochastic control scheme employing LPF-based defuzzification.

frequency-based LPFs) ensured the desired linear position with the same dynamics as the reference FLC-based scheme. However, the impact of both f_0 and f_{inf} can be clearly observed in the third row, which highlights the resulting pendulum angle for difference parameter sets. The red curve corresponds to the $f_0 = 1.6\ Hz$ cut-off frequency LPF and $f_{inf} = 5\ kHz$ configuration, where the phase lag resulted in notable increase in the pendulum angle compared to the reference fuzzy performance, i.e., the LPF increased the overshoot to 10 *deg*, while the reference fuzzy control ensured 5 *deg* IB angle overshoot. Indeed, this LPF configuration resulted in PM=40 *deg* and was the boundary condition for acceptable stabilization in our application, as it was discussed in subsections V-B and VI-C. Nevertheless, even the PM=40 *deg* along with $f_{inf} = 5\ kHz$ configuration ensured satisfying control performance, since the reference tracking was not influenced, moreover, the pendulum angle was effectively stabilized without oscillatory behavior. Less PM decrease was produced by the $f_0 = 6\ Hz$ LPF configuration, see the yellow curve for $f_{inf} = 5\ kHz$ and the green curve for $f_{inf} = 50\ kHz$ in the third row. This configuration confirmed our earlier discussion about selecting the suitable cut-off frequency for the LPF-based defuzzification process. As it can be seen, the overshoot for θ_3 was increased by only 1 *deg* and the system states settled quickly to the reference system states (see the blue curves). This experiment highlights that equivalent system states are achieved with LPF-based defuzzification, therefore the SFLCs are accurate and well-founded. It is also noticeable, that the output ripple was completely suppressed in case of $f_{inf} = 50\ kHz$ (see the green curve), however even the $f_{inf} = 5\ kHz$ inference frequency ensured satisfying control performance with minor ripple magnitude (see the yellow curve).

It can be concluded based on the simulation results that both the counter-based and LPF-based SFLC architectures satisfy the control requirements and perform both the balance and stabilization of the system. These control

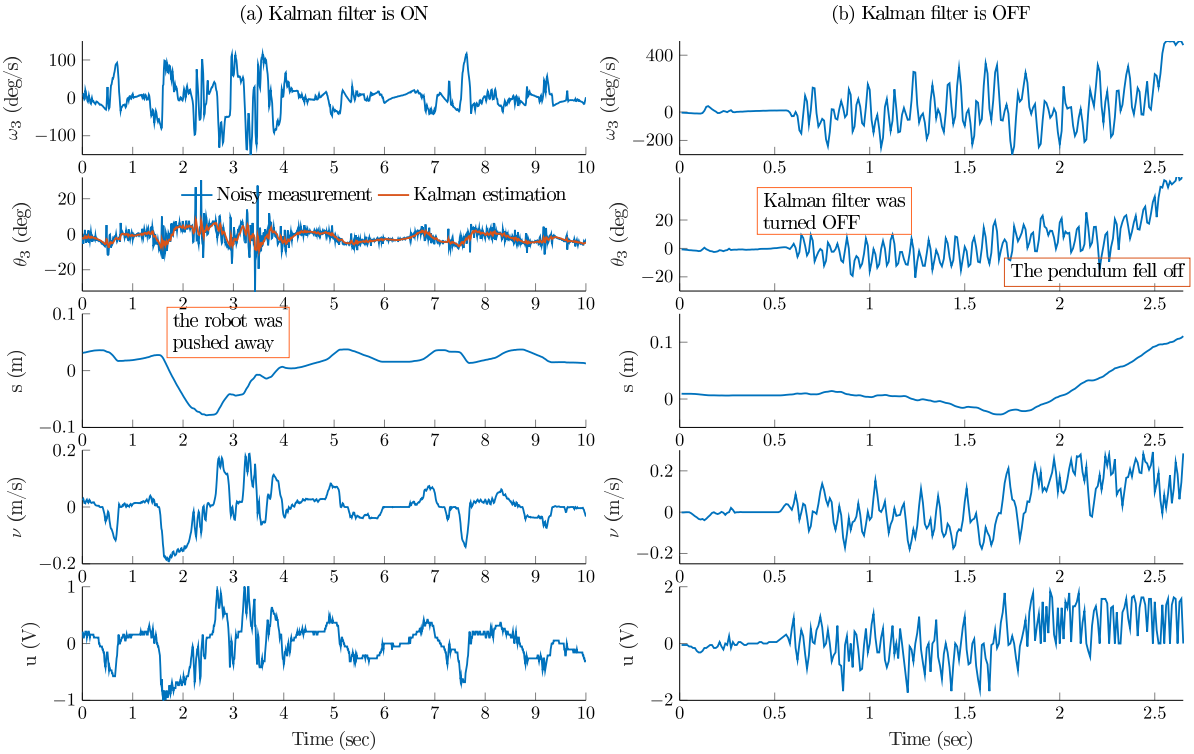


FIGURE 21. Control performance of the conventional fuzzy control scheme: (a) Kalman filter was turned on ($\hat{\theta}_3$ was fed back), the external disturbance is indicated with the framed text, (b) the Kalman estimation was turned off (θ_3^{raw} was employed in the feedback loop).

performances are compared to the conventional FLC-based control performance in Figs. 19 and 20, and it can be observed that the proposed stochastic approximation can compete with the standard fuzzy approach. Based on the aforementioned results, it is also obvious that the counter-based defuzzification process is more sensitive to decimation ratio N, i.e., the higher the inference frequency f_{inf} the more accurate crisp output approximation is provided. If the decimation ratio N is small, then the control signal is characterized by reduced resolution, which can induce oscillation in system states. On the other hand, the LPF-based defuzzification in the SFLC architecture provides less impact on the system behavior. Namely, the smoothed control action results in both oscillation-free and robust system behavior. The cost of this approach is a minor PM decrease, however as the simulation results show, there is no significant impact on the closed-loop performance in case of an adequate cut-off frequency.

VIII. EXPERIMENTAL RESULTS ON THE REAL SYSTEM

The control algorithms were implemented in a C language-based embedded software environment, where floating point calculations were applied to determine the instantaneous system state values θ_3, ω_3, s and ν.

As it was already mentioned, MCU2 works as an IMU, therefore it supplies the measurements θ_3^{raw}, ω_3, and $\hat{\theta}_3$ to MCU1 via SPI communication bus, where θ_3^{raw} denotes the IB orientation measured by the accelerometer (i.e., pendulum angle determined based on the pure accelerations), ω_3 indicates the angular velocity of the pendulum measured by the gyroscope, while $\hat{\theta}_3$ is the Kalman filter-based estimation of the IB orientation. Both the derivation of these quantities and Kalman filter implementation is described in detail in [66]. MCU1 collects the measurements at fixed $f_s = 100\ Hz$ sampling rate, moreover, it obtains the instantaneous robot position and velocity data (s and ν) based on the incremental encoder measurements in each sampling epoch. Then, it executes the chosen control algorithm based on the collected measurements (see the control schemes in Figs. 7 and 25). The crisp control signal drives the DC motors via H-bridges. Furthermore, the measurements are sent to a MATLAB GUI through the wireless module, which records the measurements. In our experiments the control frequency was set to $f_{con} = f_s = 100\ Hz$. Moreover, $f_0 = 6\ Hz$ cut-off frequency was employed in LPF-based defuzzification based on the earlier frequency domain analysis. The inference frequency was fixed to $f_{inf} = 5\ kHz$.

First the conventional fuzzy performance (Fig. 21) is addressed, then the performances of different stochastic architectures are analyzed (Figs. 22, 23 and 24). The control performances were tested in different environments, i.e., both static and dynamic behaviors of the proposed control algorithms were evaluated. During the dynamic behavior tests, the robot was pushed away from its equilibrium position several times. Moreover, the robustness against uncertain measurements was also inspected by turning on and off the Kalman estimation (if the Kalman filter was turned on, then

$\hat{\theta}_3$ was fed back to the FLCs, otherwise θ_3^{raw} was employed in the feedback loop). The experimental results highlight the raw pendulum attitude θ_3^{raw} and its estimate $\hat{\theta}_3$, the angular velocity during the stabilization ω_3, the linear displacement s and velocity v of the robot, and the applied crisp control signal u.

A. CONVENTIONAL FUZZY APPROACH

The implementation of the standard FLCs described in section V was based on the fuzzy surfaces. Since fuzzy surfaces define the crisp output as a function of the given inputs, therefore FLCs can be described with simple look-up tables (LUTs). The LUT is obtained by evaluating the possible input combinations and registering the crisp output in the table. The LUT generation process is described in detail in [61]. In our application the input ranges were partitioned into $n = 40$ evenly spaced points, therefore 40×40 size LUTs were stored in the flash memory of MCU1. The proposed scheme in Fig. 7 was executed based on the instantaneous error signals e_s, e_v, e_{θ_3} and e_{ω_3}. Once the error signals were obtained, the crisp output was calculated as follows:

$$\begin{aligned} u_s &= \mathrm{L}_{s,ij}, \quad i = *\frac{n}{2}\left(\frac{e_s}{s_m}+1\right), \quad j = *\frac{n}{2}\left(\frac{e_v}{v_m}+1\right), \\ u_{\theta_3} &= \mathrm{L}_{\theta_3,ij}, \quad i = *\frac{n}{2}\left(\frac{e_{\theta_3}}{\theta_{3m}}+1\right), \quad j = *\frac{n}{2}\left(\frac{e_{\omega_3}}{\omega_{3m}}+1\right), \\ u &= u_s + u_{\theta_3}, \end{aligned} \tag{29}$$

where $\mathrm{L}_{s,ij}$ and $\mathrm{L}_{\theta_3,ij}$ are the $n \times n$ matrices for FLC1 and FLC2, respectively. Moreover, $i = 1, \ldots, n$ and $j = 1, \ldots, n$ denote the row and column indexes, and $*$ indicates the rounding to the nearest integer (see the derivation of Eq. (29) in [61], [65]). Therefore, once the indexes were calculated, the crisp output was selected in the corresponding LUT.

The control performance of the implemented reference control scheme is shown in Fig. 21. Two parts of the whole experiment are highlighted, in which both the control quality and stabilization performance can be observed. The left side of Fig. 21 depicts the dynamics of the closed-loop system when the Kalman filter-based estimation $\hat{\theta}_3$ was employed in the regularization process. It can be seen that the standard fuzzy approach ensured acceptable performance even if external perturbation was executed. Namely, around 1.5 s the robot was pushed about 0.1 m away and it maintained both the balance of the IB and stabilization of position, moreover, the robot returned to the desired state. The robustness against measurement uncertainties was also inspected in the experiment. Therefore, the right side of Fig. 21 indicates the case when the pure accelerometer-based attitude measurement θ_3^{raw} was employed in the control loop, i.e., the Kalman filter was turned off. This experiment enabled to analyze the robustness of the control loop against uncertain data, since the attitude realization θ_3^{raw} was manipulated by external accelerations by superimposing additional noise components on the pure observation (i.e., bigger noise magnitude characterized the measurements). It can be observed, that these noisy observations and uncertain measurements highly influenced the performance of the standard fuzzy approach. Namely, the experimental results show that in approximately 2 s the robot left the vicinity of equilibrium and the pendulum fell off (see the measurements at around 2.5 s in the right side of Fig. 21).

B. STOCHASTIC FUZZY APPROACH

The proposed SFLC architecture was implemented in the embedded microcontroller as a simple C function. Let $\kappa = \{e_s, e_v, e_{\theta_3}, e_{\omega_3}\}$ denote the set of inputs of the control scheme. Then, based on Eq. (12), the stochastic fuzzification is defined for each input κ as follows:

$$\begin{aligned} \chi_{Z_\alpha}(\kappa) = 1 \Leftrightarrow -\frac{1}{2} \le \frac{\kappa}{\kappa_m} - \left(\alpha - \frac{1}{2}\right) \\ \wedge \frac{\kappa}{\kappa_m} + \left(\alpha - \frac{1}{2}\right) \le \frac{1}{2}, \end{aligned} \tag{30}$$

$$\chi_{N_\alpha}(\kappa) = 1 \Leftrightarrow \frac{\kappa}{\kappa_m} + \left(\alpha - \frac{1}{2}\right) \le -\frac{1}{2}, \tag{31}$$

$$\chi_{P_\alpha}(\kappa) = 1 \Leftrightarrow \frac{\kappa}{\kappa_m} - \left(\alpha - \frac{1}{2}\right) \ge \frac{1}{2}, \tag{32}$$

where κ/κ_m denotes the normalized input variables on their ranges, i.e., $\kappa/\kappa_m = \{\frac{e_s}{s_m}, \frac{e_v}{v_m}, \frac{e_{\theta_3}}{\theta_{3m}}, \frac{e_{\omega_3}}{\omega_{3m}}\}$ (see Fig. 8). The error signals e_s, e_v, e_{θ_3} and e_{ω_3} were obtained in each $T_s = 1/f_s$ sampling epoch based on the measurements, and the quotient κ/κ_m was updated. The fuzzification was performed at f_{inf} frequency by the Eqs. (30), (31) and (32). These equations supplied the stochastic bit streams that represent the membership function values for the linguistic variables N, Z, and P for each error signal (see the stochastic fuzzifiers in Fig. 25). The inference machine was composed of 6 AND operations, while the aggregation process employed 2 OR operations to represent the output membership function values u_P and u_N for each controller. For example the characteristic functions $\chi_{P_\alpha}(u_s)$ and $\chi_{N_\alpha}(u_s)$ of the α-cut set of fuzzy sets P and N for SFLC1 were obtained at f_{inf} frequency as given in Eq. (33), as shown at the bottom of the page, (see the inference machine in Fig. 25).

Finally, the aforementioned aggregated stochastic sequences $\chi_{P_\alpha}(u_s)$, $\chi_{N_\alpha}(u_s)$, $\chi_{P_\alpha}(u_{\theta_3})$ and $\chi_{N_\alpha}(u_{\theta_3})$ were supplied to the counter-based or LPF-based defuzzification

$$\begin{aligned} \chi_{P_\alpha}(u_s) &= \left(\chi_{N_\alpha}(e_{\theta_3}) \wedge \chi_{N_\alpha}(e_{\omega_3})\right) \vee \left(\chi_{Z_\alpha}(e_{\theta_3}) \wedge \chi_{N_\alpha}(e_{\omega_3})\right) \vee \left(\chi_{N_\alpha}(e_{\theta_3}) \wedge \chi_{Z_\alpha}(e_{\omega_3})\right), \\ \chi_{N_\alpha}(u_s) &= \left(\chi_{P_\alpha}(e_{\theta_3}) \wedge \chi_{Z_\alpha}(e_{\omega_3})\right) \vee \left(\chi_{Z_\alpha}(e_{\theta_3}) \wedge \chi_{P_\alpha}(e_{\omega_3})\right) \vee \left(\chi_{P_\alpha}(e_{\theta_3}) \wedge \chi_{P_\alpha}(e_{\omega_3})\right), \end{aligned} \tag{33}$$

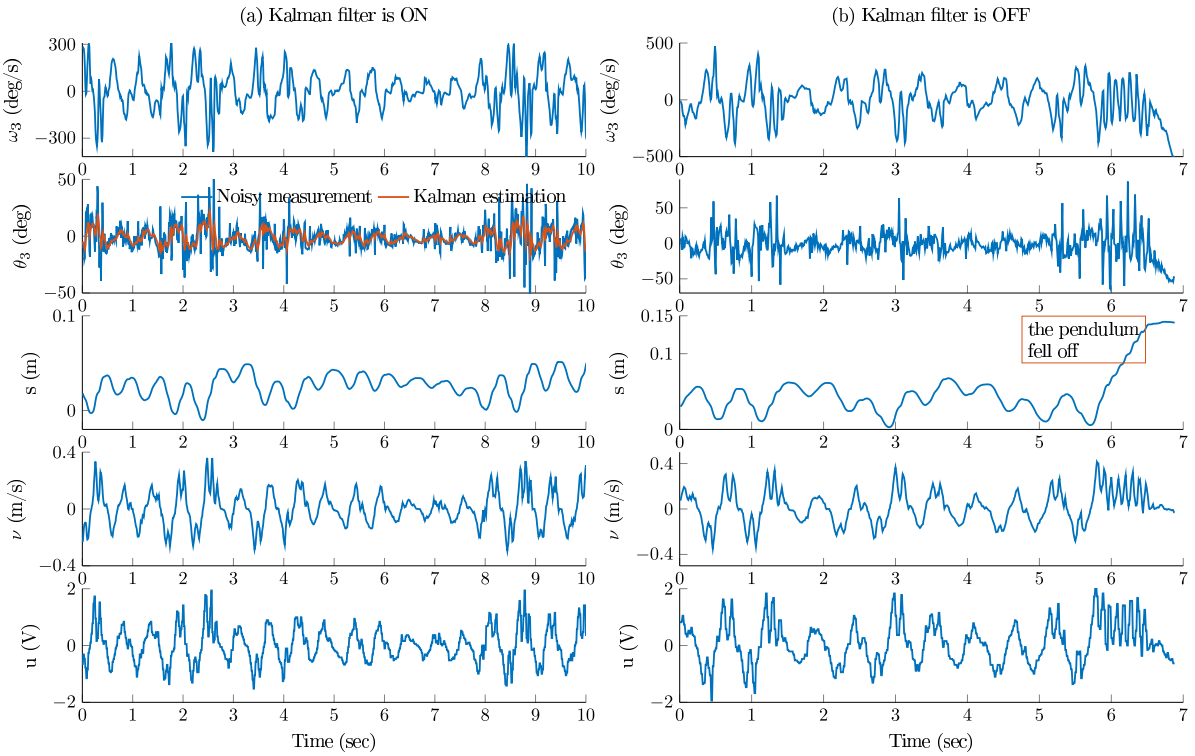


FIGURE 22. Control performance of counter defuzzification unit-based stochastic fuzzy approach: (a) Kalman filter was turned on ($\hat{\theta}_3$ was fed back), (b) the Kalman estimation was turned off (θ_3^{raw} was employed in the feedback loop).

units at f_{inf} frequency, which provided the stochastic approximations of crisp control signals u_s and u_{θ_3}. The uniform random numbers were generated with a 16 bit LFSR. Both the counter-based and LPF-based defuzzification methods were tested on the real system. This equivalent SC-based fuzzy approach was evaluated in four experiments. The experimental results are discussed as follows.

1) SFLC PERFORMANCE WITH COUNTER-BASED DEFUZZIFICATION

Fig. 22 highlights two parts of the experiment, where the control performance of the counter defuzzification unit-based SFLC scheme is presented. Similarly to the standard fuzzy approach, the effects of both external disturbances and uncertain measurements were tested to inspect the robustness of the control scheme. The left side of the figure shows the experimental results for the case when the Kalman filter was turned on, i.e., the smoothed $\hat{\theta}_3$ was fed back to the controller. It can be observed, that the control scheme stabilized the system (both the position and pendulum angle), however quite modest performance was achieved with $f_{inf} = 5\ kHz$ inference speed. Namely, the system was successfully stabilized in the vicinity of equilibrium, however the states were characterized by significant oscillations. This outcome was expected, since as was discussed in section VII, the low decimation ratio results in reduced resolution in the control signal. This reduced resolution gives rise to an oscillating error signal around the desired equilibrium, since even if

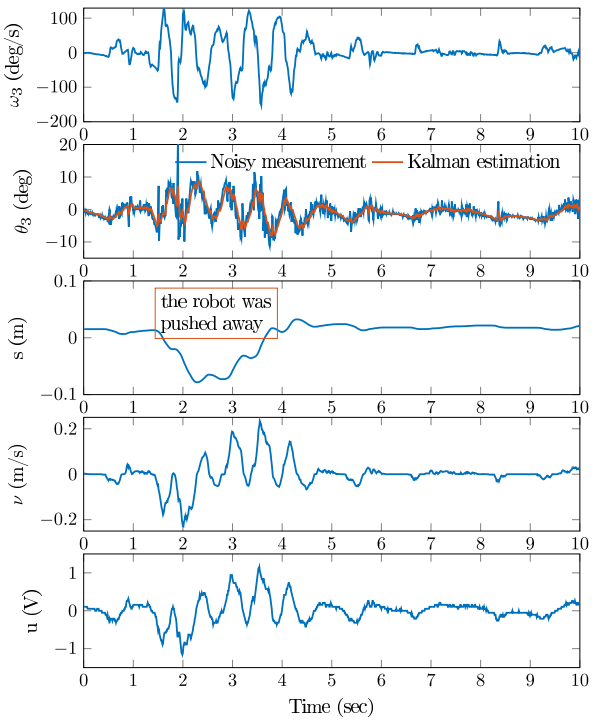


FIGURE 23. Control performance of LPF defuzzification unit-based stochastic fuzzy approach. The Kalman was turned on ($\hat{\theta}_3$ was fed back).

the system states are close to the desired set point the crisp output will lead the states to the vicinity with the smallest incremental change in the control signal.

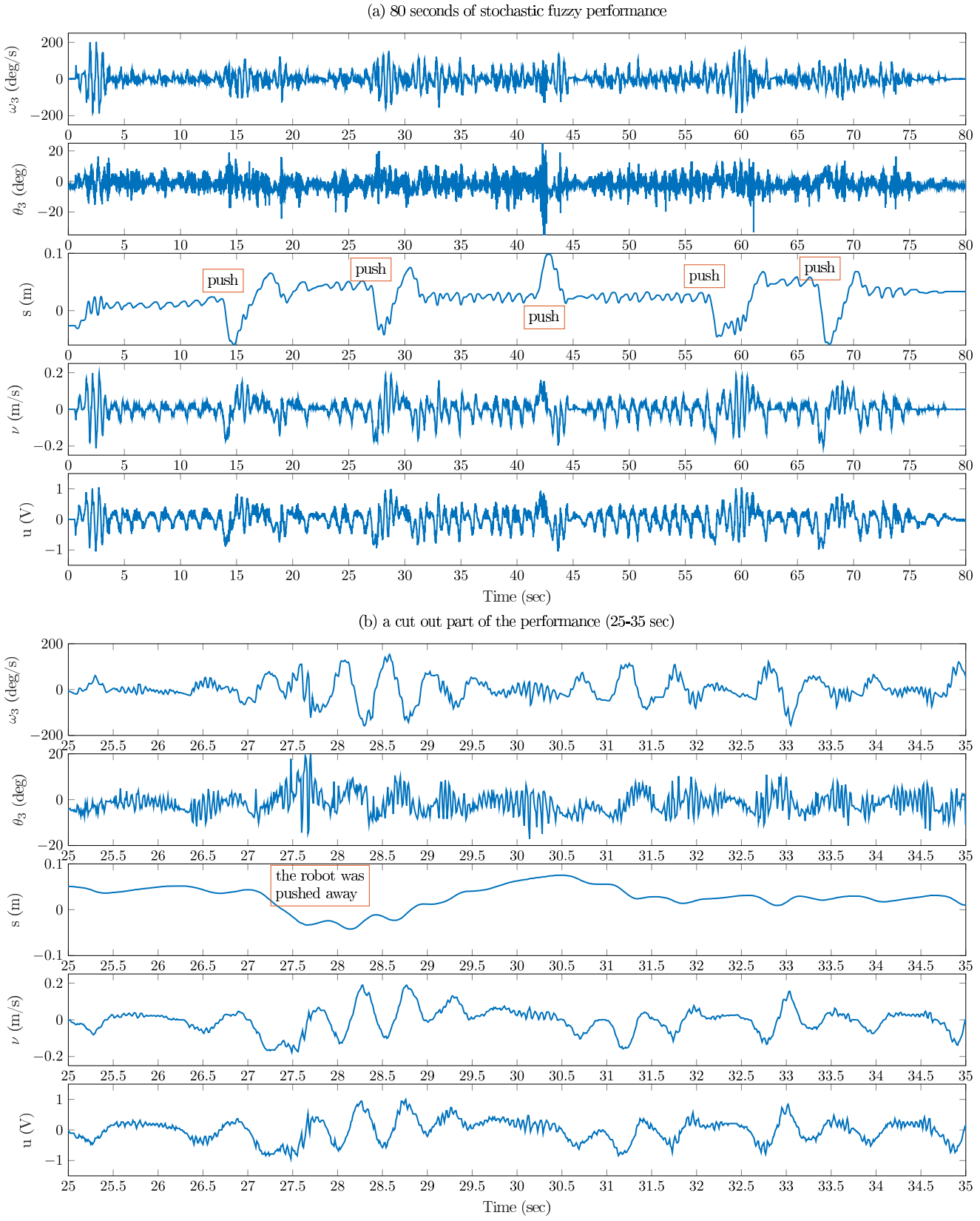


FIGURE 24. Control performance of LPF defuzzification unit-based stochastic fuzzy approach. The Kalman estimation was turned off: (a) 80 seconds of control performance is presented, (b) 10 seconds more detailed illustration.

The right side of Fig. 22 depicts the achieved control performance, when the Kalman filter was turned off (i.e., the noisy θ_3^{raw} was employed in the feedback loop). It is highlighted that the control scheme was not able to handle the noisy measurements robustly, i.e., the pendulum fell off after 6 s of regulation. It can be concluded that the small decimation ratio in the counter-based defuzzification unit resulted in poor control performance in steady state conditions, moreover, the performance was unacceptable in dynamic environment when increased noise magnitudes characterized the system states (see the right side of Fig. 22). The control signal resolution can be increased with higher decimation ratio, e.g., the simulation results showed that $f_{inf} = 50\ kHz$ provided smooth control action and satisfying control performance (see Fig. 19). However, the modest resources of the embedded microcontroller limited the experimental verification to only 5 kHz inference speed in our application, therefore the performance evaluation of $f_{inf} = 50\ kHz$ is left open for investigation in future studies.

2) SFLC PERFORMANCE WITH LPF-BASED DEFUZZIFICATION

The stochastic control scheme, which employed LPF defuzzification unit-based SFLCs, showed remarkably satisfying control performance. On one hand, the closed-loop structure provided robustness against external disturbances. It can be seen in Fig. 23, that the robot was pushed away from its equilibrium state around at 1.5 s, and the stochastic control structure was able to both stabilize the robot and drive the system states back to the desired set point. In this experiment the Kalman filter was turned on, therefore the estimated IB orientation $\hat{\theta}_3$ was fed back to the SFLC. It is also shown that smooth control action was provided by the proposed stochastic architecture, since the desired set points (both robot position and pendulum angle) were maintained efficiently without any oscillatory effects.

The robustness against measurement uncertainties was also investigated in the LPF-based SFLC architecture. Fig. 24 highlights that the stochastic scheme provided remarkable control performance, even when the raw and noisy θ_3^{raw} measurements were fed back to the SFLC (i.e., the Kalman filter was turned off). The control architecture was capable of supplying smooth control signals even in this noisy environment, moreover, it both effectively stabilized the robot position and kept the pendulum in the vertical upright position. Additionally to supplying noisy system states, the robot was pushed away several times and the stochastic controllers robustly handled the disturbances and efficiently drove back the robot to the equilibrium. Fig. 24 shows 80 s of the experiment, where despite the external disturbances and noisy environment the stabilization was successful, robust and reliable.

It can be concluded that the implemented LPF-based SFLCs provide smooth control action and are characterized by robust performance. The experimental results indicate that the flexibility of the proposed stochastic architecture enabled it to compete and even outperform the equivalent standard (FLC-based) control approach. These results were also observed in the simulation results in Fig. 20, where despite the small PM decrease, smooth control action was provided for the stabilization of the robot. The LFP-based defuzzification stage is a more flexible structure than the up-down counter-based approach, since the filter can be tuned to the system dynamics (see Eq. (27)). It has also been proven, that the frequency domain calculations (discussed in subsection V-B) provided good basis for stochastic control design, since satisfying closed-loop dynamics was achieved based on the obtained results. Namely, the $f_0 = 6\ Hz$ cut-off frequency did not influence the dynamics of the closed-loop system.

IX. CONCLUSION

This paper introduced a SL-based fuzzy controller (i.e., SFLCs) as a new fuzzy technology to establish heuristic inference machines. The proposed architecture was experimentally validated in a control scheme for the stabilization of a real SBR. First, the conventional SL and relevant stochastic fuzzy elements were introduced. Next, the SBR (as the plant to be controlled) was described and its mathematical model was presented. Thereafter, two control methods were designed to stabilize the system. Namely, a conventional FLC-based control approach as a reference scheme was utilized for the stabilization of the plant first. Then, based on both the reference scheme and FLC parameters, the equivalent SFLC-based architecture was established, which is one of the main results of this article. The whole SFLC architecture from fuzzification, over the inference system, to defuzzification was derived in detail. The effectiveness and robustness of the proposed control method were proven with both simulation and experimental results. It was found that the performance of SFLC architecture is determined by two parameters. Namely, the ratio of inference and control frequencies determines the resolution of the crisp output in case of counter-based defuzzification. This decimation ratio should be defined based on both the control requirements and plant sensitivity in order to suppress the oscillatory behavior. On the other hand, the SFLC performance is determined by the inference and cut-off frequencies in case of LPF-based defuzzification. In this approach, the LPF should not alter the stability margins significantly, while the inference frequency can be increased to reduce the output ripple of the crisp output. In our experiments, the LPF-based defuzzification method showed extremely good performance during the SFLC-based control, where despite both the uncertain measurements and external disturbances, successful stabilization and robust control performance were achieved. The proposed SFLC-based control method outperformed the equivalent conventional FLC-based approach in terms of robustness against uncertain measurements. Moreover, despite the hardware limitations of the SBR competitive control performance was achieved regarding the stochastic fuzzy control. Since the applied SBR was characterized by modest resources and was not specifically built for the purpose of testing the proposed stochastic architecture, therefore the future work is focused on conducting experiments on a more powerful hardware.

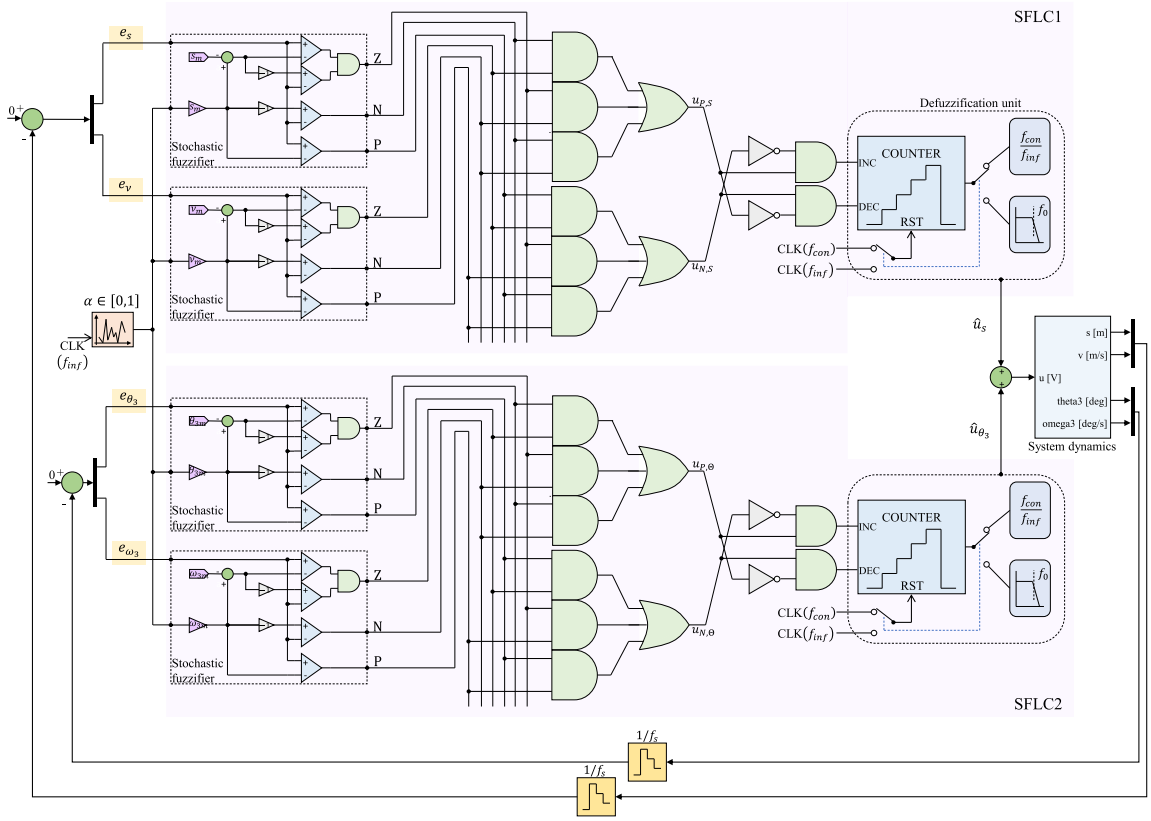


FIGURE 25. The equivalent SC-based complete fuzzy control scheme.

We intend to analyze the approximation errors and robustness indexes as a function of both increased inference frequencies and more advanced filter implementations.

APPENDIX A
SC-BASED FUZZY CONTROL SCHEME

Fig. 25 shows the SC-based two-loop fuzzy control scheme for the stabilization of the plant, where the reference controllers (i.e., both FLC1 and FLC2) have been approximated with their stochastic counterparts (SFLC1 and SFLC2). Based on the stochastic structure introduced in Fig. 12 the definition of SFLC2 was straightforward. The error signals (e_{θ_3}, e_{ω_3}) and the random number generator drive the stochastic structure, whose output u_{θ_3} is obtained based on the aggregated stochastic bit streams $u_{P,\Theta}$ and $u_{N,\Theta}$ that represent the consequent fuzzy sets in stochastic domain. The fuzzification state employs the fuzzy membership function parameters θ_{3m} and ω_{3m} (see Fig. 8) for the generation of the Z, N and P stochastic bit streams. In addition, the same random number generator was applied for both SFLC1 and SFLC2 because the controllers are independent.

APPENDIX B
EFFECTIVE BIT DEPTH RESULTS FOR COUNTER-BASED DEFUZZIFICATION

Recall that single-precision floating-point format uses 32 bits to represent the crisp quantity, where the first bit denotes the sign, the next 8 bits represent the exponent, and the remaining 23 bits are used for the fraction bits (i.e., significand precision). Fig. 26 highlights the achieved effective bits numbers L in stochastic approximation that are consistent with the reference crisp output according to IEEE 754 standard. It can be observed that a magnitude increase in the decimation ratio results in a binary digit increase in the mean value of the representation. For example in case of $f_{inf}/f_{con} = 5000$ (see the bottom-left plot of Fig. 26), the mean value μ_L of bit depth L is approximately 14 bits, i.e., the sign and exponent are represented with full precision, while 5 fraction

TABLE 6. Number of arithmetic operations for different architectures.

Conventional FLC

Fuzzification	+ 3M + 3A + 2C + 1L + 2B
Number of inputs	×2
Fuzzy operator	+ C
Rule base	×9
Defuzzification	+ 6C +3M +2A
Sum	**= 56A + 36B + 51C + 18L + 57M**

Stochastic FLC

Fuzzification	+ 4C + 2A + 1L	
Number of inputs	×2	
Fuzzy operator and rule base	+ 9L	
Defuzzification	Counter-based	LPF-based
	+ 6L + 1A + 1M	+ 6L + 2A + 2M
Sum	**= 5A + 8C + 17L + M**	**= 6A + 8C + 17L + 2M**

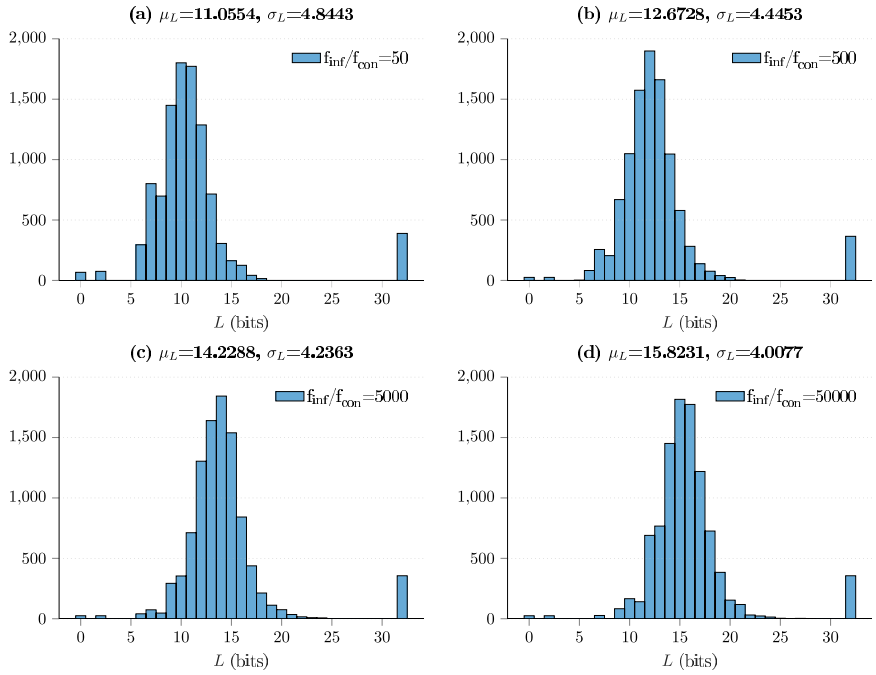


FIGURE 26. The achievable bit depth for different decimation ratios (*N*): (a) $N = 50$, (b) $N = 500$, (c) $N = 5000$ and (d) $N = 50000$.

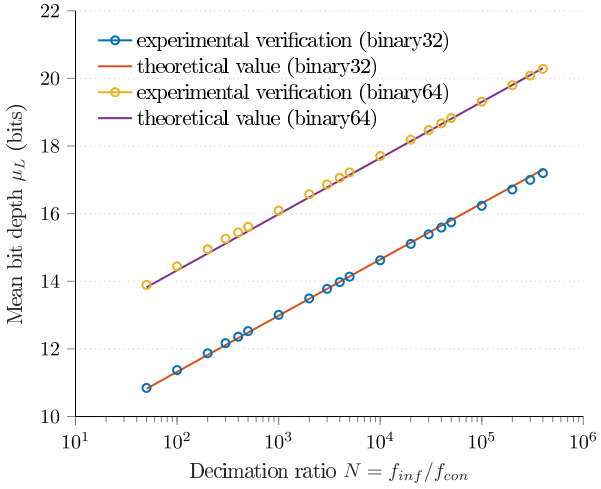


FIGURE 27. The achievable mean bit depth for different decimation ratios (*N*).

bits are consistent (exactly same) with the reference crisp output representation. Similar analyses can be conducted for 64-bit double-precision floating-point format, where 12 bits are used for the sign and exponent representation, and the rest of the bits denotes the significand precision.

APPENDIX C
COMPUTATIONAL COMPLEXITIES

The computation burden of the proposed stochastic controller (SFLC) compared to the conventional FLC in a standard microcontroller-based implementation is discussed as follows. Let M, A and C denote the IEEE 754 standard floating-point multiplication, addition and comparison, respectively (32-bit or 64-bit, which depends on the employed architecture). Moreover, let L represent the Boolean logical operation and B denote the number of branches in the implementation. Then, the arithmetic operations employed in both control algorithms are summarized in Table 6. The difference in computational complexities is significant, however the requirement of higher inference speed should be noted in case of the SFLC architecture. Hence, the real architectural advantage of SFLCs is exploited if simple hardware elements implement the complete fuzzy inference method in FPGAs. This analysis forms the next step in our future study.

REFERENCES

[1] Z. Qin, Y. Qiu, M. Zheng, H. Dong, Z. Lu, Z. Wang, and H. Pan, "A universal approximation method and optimized hardware architectures for arithmetic functions based on stochastic computing," *IEEE Access*, vol. 8, pp. 46229–46241, 2020.

[2] M. Alawad and M. Lin, "Survey of stochastic-based computation paradigms," *IEEE Trans. Emerg. Topics Comput.*, vol. 7, no. 1, pp. 98–114, Jan. 2019.

[3] T. Figliolia and A. G. Andreou, "An FPGA multiprocessor architecture for Bayesian online change point detection using stochastic computation," *Microprocessors Microsyst.*, vol. 74, Apr. 2020, Art. no. 102968.

[4] P. Li, D. J. Lilja, W. Qian, K. Bazargan, and M. D. Riedel, "Computation on stochastic bit streams digital image processing case studies," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 22, no. 3, pp. 449–462, Mar. 2014.

[5] M. H. Najafi and M. E. Salehi, "A fast fault-tolerant architecture for sauvola local image thresholding algorithm using stochastic computing," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 24, no. 2, pp. 808–812, Feb. 2016.

[6] N. Onizawa, D. Katagiri, K. Matsumiya, W. J. Gross, and T. Hanyu, "An accuracy/energy-flexible configurable Gabor-filter chip based on stochastic computation with dynamic voltage–frequency–length scaling," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 8, no. 3, pp. 444–453, Sep. 2018.

[7] R. Wang, J. Han, B. F. Cockburn, and D. G. Elliott, "Design, evaluation and fault-tolerance analysis of stochastic FIR filters," *Microelectron. Rel.*, vol. 57, pp. 111–127, Feb. 2016.

[8] Y. Liu and K. K. Parhi, "Linear-phase lattice FIR digital filter architectures using stochastic logic," *J. Signal Process. Syst.*, vol. 90, no. 5, pp. 791–803, May 2018.

[9] Y. Liu and K. K. Parhi, "Architectures for recursive digital filters using stochastic computing," *IEEE Trans. Signal Process.*, vol. 64, no. 14, pp. 3705–3718, Jul. 2016.

[10] H. Ichihara, T. Sugino, S. Ishii, T. Iwagaki, and T. Inoue, "Compact and accurate digital filters based on stochastic computing," *IEEE Trans. Emerg. Topics Comput.*, vol. 7, no. 1, pp. 31–43, Jan. 2019.

[11] V. T. Lee, A. Alaghi, R. Pamula, V. S. Sathe, L. Ceze, and M. Oskin, "Architecture considerations for stochastic computing accelerators," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 37, no. 11, pp. 2277–2289, Nov. 2018.

[12] B. Li, M. H. Najafi, and D. J. Lilja, "Low-cost stochastic hybrid multiplier for quantized neural networks," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 15, no. 2, pp. 1–19, Jun. 2019.

[13] M. Lunglmayr, D. Wiesinger, and W. Haselmayr, "Design and analysis of efficient maximum/minimum circuits for stochastic computing," *IEEE Trans. Comput.*, vol. 69, no. 3, pp. 402–409, Mar. 2020.

[14] S. Liu, W. J. Gross, and J. Han, "Introduction to dynamic stochastic computing," *IEEE Circuits Syst. Mag.*, vol. 20, no. 3, pp. 19–33, 3rd Quart., 2020.

[15] W. Haselmayr, D. Wiesinger, and M. Lunglmayr, "High-accuracy and fault tolerant stochastic inner product design," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 67, no. 3, pp. 541–545, Mar. 2020.

[16] A. Alaghi, W. Qian, and J. P. Hayes, "The promise and challenge of stochastic computing," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 37, no. 8, pp. 1515–1531, Aug. 2018.

[17] B. R. Gaines, "Stochastic computer thrives on noise," *Electronics*, vol. 40, no. 14, p. 72, 1967.

[18] B. R. Gaines, "Stochastic computing systems," in *Advances in Information Systems Science*. Boston, MA, USA: Springer, 1969, pp. 37–172.

[19] Q. Zhang, M. Zhang, T. Chen, Z. Sun, Y. Ma, and B. Yu, "Recent advances in convolutional neural network acceleration," *Neurocomputing*, vol. 323, pp. 37–51, Jan. 2019.

[20] H. Li, R. Cai, N. Liu, X. Lin, and Y. Wang, "Deep reinforcement learning: Algorithm, applications, and ultra-low-power implementation," *Nano Commun. Netw.*, vol. 16, pp. 81–90, Jun. 2018.

[21] Y. Kondo and Y. Sawada, "Functional abilities of a stochastic logic neural network," *IEEE Trans. Neural Netw.*, vol. 3, no. 3, pp. 434–443, May 1992.

[22] S. L. Bade and B. L. Hutchings, "FPGA-based stochastic neural networks-implementation," in *Proc. IEEE Workshop FPGAs for Custom Comput. Mach.*, Apr. 1994, pp. 189–198.

[23] E. M. Petriu, K. Watanabe, and T. H. Yeap, "Applications of random-pulse machine concept to neural network design," *IEEE Trans. Instrum. Meas.*, vol. 45, no. 2, pp. 665–669, Apr. 1996.

[24] B. D. Brown and H. C. Card, "Stochastic neural computation. I. Computational elements," *IEEE Trans. Comput.*, vol. 50, no. 9, pp. 891–905, Sep. 2001.

[25] A. Farkas, G. Kertesz, and R. Lovas, "Parallel and distributed training of deep neural networks: A brief overview," in *Proc. IEEE 24th Int. Conf. Intell. Eng. Syst. (INES)*, Jul. 2020, pp. 165–170.

[26] G. Kertesz, S. Szenasi, and Z. Vamossy, "Parallelization methods of the template matching method on graphics accelerators," in *Proc. 16th IEEE Int. Symp. Comput. Intell. Informat. (CINTI)*, Nov. 2015, pp. 161–164.

[27] A. Ardakani, F. Leduc-Primeau, N. Onizawa, T. Hanyu, and W. J. Gross, "VLSI implementation of deep neural network using integral stochastic computing," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 10, pp. 2688–2699, Oct. 2017.

[28] A. Ren, Z. Li, C. Ding, Q. Qiu, Y. Wang, J. Li, X. Qian, and B. Yuan, "SC-DCNN: Highly-scalable deep convolutional neural network using stochastic computing," *ACM SIGPLAN Notices*, vol. 52, no. 4, pp. 405–418, 2017.

[29] D. Zhang and H. Li, "A stochastic-based FPGA controller for an induction motor drive with integrated neural network algorithms," *IEEE Trans. Ind. Electron.*, vol. 55, no. 2, pp. 551–561, Feb. 2008.

[30] H. Li, D. Zhang, and S. Y. Foo, "A stochastic digital implementation of a neural network controller for small wind turbine systems," *IEEE Trans. Power Electron.*, vol. 21, no. 5, pp. 1502–1507, Sep. 2006.

[31] Y. Liu, S. Liu, Y. Wang, F. Lombardi, and J. Han, "A stochastic computational multi-layer perceptron with backward propagation," *IEEE Trans. Comput.*, vol. 67, no. 9, pp. 1273–1286, Sep. 2018.

[32] Y. Liu, L. Liu, F. Lombardi, and J. Han, "An energy-efficient and noise-tolerant recurrent neural network using stochastic computing," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 27, no. 9, pp. 2213–2221, Sep. 2019.

[33] X. Liu and K. K. Parhi, "Molecular and DNA artificial neural networks via fractional coding," *IEEE Trans. Biomed. Circuits Syst.*, vol. 14, no. 3, pp. 490–503, Jun. 2020.

[34] V. Canals, A. Morro, A. Oliver, M. L. Alomar, and J. L. Rossello, "A new stochastic computing methodology for efficient neural network implementation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 3, pp. 551–564, Mar. 2016.

[35] H. Sim and J. Lee, "Cost-effective stochastic MAC circuits for deep neural networks," *Neural Netw.*, vol. 117, pp. 152–162, Sep. 2019.

[36] J. Li, Z. Yuan, Z. Li, A. Ren, C. Ding, J. Draper, S. Nazarian, Q. Qiu, B. Yuan, and Y. Wang, "Normalization and dropout for stochastic computing-based deep convolutional neural networks," *Integration*, vol. 65, pp. 395–403, Mar. 2019.

[37] Y. Liu, S. Liu, Y. Wang, F. Lombardi, and J. Han, "A survey of stochastic computing neural networks for machine learning applications," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Aug. 5, 2020, doi: 10.1109/TNNLS.2020.3009047.

[38] P. Jeavons, D. A. Cohen, and J. Shawe-Taylor, "Generating binary sequences for stochastic computing," *IEEE Trans. Inf. Theory*, vol. 40, no. 3, pp. 716–720, May 1994.

[39] J. G. Ortega, C. L. Janer, J. M. Quero, L. G. Franquelo, J. Pinilla, and J. Serrano, "Analog to digital and digital to analog conversion based on stochastic logic," in *Proc. 21st Annu. Conf. IEEE Ind. Electron. (IECON)*, vol. 2, Nov. 1995, pp. 995–999.

[40] C. L. Janer, J. M. Quero, J. G. Ortega, and L. G. Franquelo, "Fully parallel stochastic computation architecture," *IEEE Trans. Signal Process.*, vol. 44, no. 8, pp. 2110–2117, Aug. 1996.

[41] S. L. Toral, J. M. Quero, and L. G. Franquelo, "Stochastic pulse coded arithmetic," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, vol. 1, May 2000, pp. 599–602.

[42] V. V. Vujicic, S. S. Milovancev, M. D. Pesaljevic, D. V. Pejic, and I. Z. Zupunski, "Low-frequency stochastic true RMS instrument," *IEEE Trans. Instrum. Meas.*, vol. 48, no. 2, pp. 467–470, Apr. 1999.

[43] V. C. Gaudet and A. C. Rapley, "Iterative decoding using stochastic computation," *IET Electron. Lett.*, vol. 39, no. 3, pp. 299–301, Apr. 2003.

[44] S. Sharifi Tehrani, W. J. Gross, and S. Mannor, "Stochastic decoding of LDPC codes," *IEEE Commun. Lett.*, vol. 10, no. 10, pp. 716–718, Oct. 2006.

[45] S. Sharifi Tehrani, S. Mannor, and W. J. Gross, "Fully parallel stochastic LDPC decoders," *IEEE Trans. Signal Process.*, vol. 56, no. 11, pp. 5692–5703, Nov. 2008.

[46] W. Qian, X. Li, M. D. Riedel, K. Bazargan, and D. J. Lilja, "An architecture for fault-tolerant computation with stochastic logic," *IEEE Trans. Comput.*, vol. 60, no. 1, pp. 93–105, Jan. 2011.

[47] A. Torralba, F. Colodro, and L. G. Franquelo, "A fuzzy-logic controller with on-chip learning, employing stochastic logic," in *Proc. IEEE 3rd Int. Fuzzy Syst. Conf.*, Jun. 1994, pp. 1759–1764.

[48] F. Colodro, A. Torralba, and L. G. Franquelo, "A digital fuzzy-logic controller with a simple architecture," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, vol. 2, May/Jun. 1994, pp. 101–104.

[49] F. Colodro, A. Torralba, R. Carvajal, and L. G. Franquelo, "A fuzzy logic controller using stochastic logic," in *Proc. IEEE Int. Symp. Circuits Syst. Circuits Syst. Inf. (Age ISCAS)*, Jun. 1997, pp. 629–632.

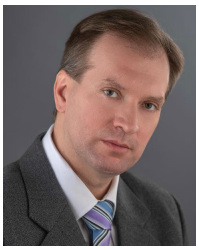
[50] S. Dick, V. Gaudet, and H. Bai, "Bit-serial arithmetic: A novel approach to fuzzy hardware implementation," in *Proc. Annu. Meeting North Amer. Fuzzy Inf. Process. Soc. (NAFIPS)*, May 2008, pp. 1–6.

[51] K. Nagy, S. Divéki, P. Odry, M. Sokola, and V. Vujičić, "A stochastic approach to fuzzy control," *Acta Polytechnica Hungarica*, vol. 9, no. 6, pp. 29–48, 2012.

[52] A. H. Zavala and O. C. Nieto, "Fuzzy hardware: A retrospective and analysis," *IEEE Trans. Fuzzy Syst.*, vol. 20, no. 4, pp. 623–635, Aug. 2012.

[53] G. Bosque, I. del Campo, and J. Echanobe, "Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades," *Eng. Appl. Artif. Intell.*, vol. 32, pp. 283–331, Jun. 2014.
[54] A. Alaghi and J. P. Hayes, "Survey of stochastic computing," *ACM Trans. Embedded Comput. Syst.*, vol. 12, no. 2s, pp. 1–19, May 2013.
[55] M. H. Najafi, D. Jenson, D. J. Lilja, and M. D. Riedel, "Performing stochastic computation deterministically," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 27, no. 12, pp. 2925–2938, Dec. 2019.
[56] G. Gyorok, "Semiconductor-based random number generator implemented with a microcontroller," in *Proc. IEEE 24th Int. Conf. Intell. Eng. Syst. (INES)*, Jul. 2020, pp. 191–196.
[57] S. A. Salehi, "Low-cost stochastic number generators for stochastic computing," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 28, no. 4, pp. 992–1001, Apr. 2020.
[58] J. Hu, B. Li, C. Ma, D. Lilja, and S. J. Koester, "Spin-hall-effect-based stochastic number generator for parallel stochastic computing," *IEEE Trans. Electron Devices*, vol. 66, no. 8, pp. 3620–3627, Aug. 2019.
[59] C. Winstead, "Tutorial on stochastic computing," in *Stochastic Computing: Techniques and Applications*. Cham, Switzerland: Springer, 2019, pp. 39–76.
[60] T.-H. Chen and J. P. Hayes, "Design of division circuits for stochastic computing," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI (ISVLSI)*, Jul. 2016, pp. 116–121.
[61] Á. Odry, R. Fullér, I. J. Rudas, and P. Odry, "Fuzzy control of self-balancing robots: A control laboratory project," *Comput. Appl. Eng. Educ.*, vol. 28, no. 4, pp. 512–535, 2020.
[62] A. Odry, "New soft computing-based methods in sensor fusion and control: Applications on a real mechatronic system," Ph.D. dissertation, School Appl. Inform. Appl. Math., John von Neumann Fac. Inform., Óbuda Univ., Budapest, Hungary, 2020.
[63] A. Odry, I. Harmati, Z. Király, and P. Odry, "Design, realization and modeling of a two-wheeled mobile pendulum system," *Proc. 14th Int. Conf. Instrum., Meas., Circuits Syst. (IMCAS)*, 2015, pp. 75–79.
[64] A. Odry, E. Burkus, I. Kecskes, J. Fodor, and P. Odry, "Fuzzy control of a two-wheeled mobile pendulum system," in *Proc. IEEE 11th Int. Symp. Appl. Comput. Intell. Informat. (SACI)*, May 2016, pp. 99–104.
[65] A. Odry, I. Kecskés, E. Burkus, and P. Odry, "Protective fuzzy control of a two-wheeled mobile pendulum robot: Design and optimization," *WSEAS Trans. Syst. Control*, vol. 12, pp. 297–306, 2017.
[66] Á. Odry, R. Fullér, I. J. Rudas, and P. Odry, "Kalman filter for mobile-robot attitude estimation: Novel optimized and adaptive solutions," *Mech. Syst. Signal Process.*, vol. 110, pp. 569–589, Sep. 2018.
[67] J.-J. E. Slotine and W. Li, *Applied Nonlinear Control*, vol. 199, no. 1. Englewood Cliffs, NJ, USA: Prentice-Hall, 1991.
[68] C.-T. Chen, *Linear System Theory and Design*. Oxford, U.K.: Oxford Univ. Press, 1998.
[69] J. H. Lilly, *Fuzzy Control and Identification*. Hoboken, NJ, USA: Wiley, 2011.
[70] G. F. Franklin, J. D. Powell, A. Emami-Naeini, and H. Sanjay, *Feedback Control of Dynamic Systems*. London, U.K.: Pearson, 2015.

ÁKOS ODRY received the B.Sc. and M.Sc. degrees in electrical engineering from the Budapest University of Technology and Economics, in 2012 and 2015, respectively, and the Ph.D. degree from the Doctoral School of Applied Informatics and Applied Mathematics, Óbuda University, in 2020. He is currently an Assistant Lecturer with the Department of Control Engineering and Information Technology, University of Dunaújváros. His research interests include nonlinear control systems, sensor fusion, robotics, engineering education, soft computing, and adaptive control methods.



VLADIMIR TADIC received the Graduate Engineer, M.Sc., and Ph.D. degrees in electrical engineering from the Faculty of Technical Sciences, University of Novi Sad, in 2004, 2009, and 2018, respectively. He is currently a Researcher with the Department of Control Engineering and Information Technology, University of Dunaújváros. His research interests include image processing, computer vision, signal processing, speech processing, telecommunications, electronics, robotics, and soft computing methods.

PETER ODRY received the M.Sc. and Ph.D. degrees in electrical engineering from the University of Belgrade, in 1986 and 1992, respectively. He is currently a Professor with the Department of Control Engineering and Information Technology, University of Dunaújváros. His research interests include impedance tomography, legged robotic systems, computer vision, robust control techniques, and soft computing methods.