IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 7, 2021, accepted April 21, 2021, date of publication April 26, 2021, date of current version May 5, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3075452

Self-Tuning Neural Network PID With Dynamic Response Control

OMAR RODRÍGUEZ-ABREO[1,2], JUVENAL RODRÍGUEZ-RESÉNDIZ[3], (Senior Member, IEEE), CARLOS FUENTES-SILVA[1], RODRIGO HERNÁNDEZ-ALVARADO[1], AND MARÍA DEL CONSUELO PATRICIA TORRES FALCÓN[1]

[1]Industrial technologies division, Universidad Politécnica de Querétaro, Santiago de Querétaro 76240, México
[2]Instituto Politécnico Nacional, Santiago de Querétaro 76090, México
[3]Faculty of Engineering, Universidad Autónoma de Querétaro, Santiago de Querétaro 76010, México

Corresponding author: Omar Rodríguez-Abreo (omar.rodriguez@upq.edu.mx)

This work was supported by the Universidad Politécnica de Querétaro, México.

ABSTRACT PID controllers are widely used and adaptable to various types of systems. However, for the response to be adequate under different conditions, the PID gains must be adjusted. The tuning is made according to the difference between the reference value and the real value (error). This work presents a self-adjusting PID controller based on a backpropagation artificial neural network. The network calculates the appropriate gains according to the desired output, that is, the dynamic response desired which is composed of the transient part and the stationary part of the step response of a system. The contribution of the work is that in addition to using the error for network training, the maximum desired values of overshoots, settling times, and stationary errors were used as input data for the network. An offline training database was created using genetic algorithms to obtain the dynamic response data associated with PID gains. The genetic algorithm allows getting data in different operating ranges and allows using only stable gains combinations. The database was used for training. Subsequently, the neural network estimates an appropriate gain combination, adapting to the error and the desired response. The method performance is evaluated by controlling the speed of a direct current motor. The results indicate an average error of 4% for the database between the requested and system response. On the other hand, the gains estimated by the network in the test dataset (1544 combinations) did not cause instability and complying with the expected dynamic response in 86% of the dataset.

INDEX TERMS Auto-tuning, speed control, genetic algorithm, neural network, PID, dynamic response.

NOMENCLATURE

PID	Proportional integral derivate controller
ANN	Artificial neural network
ts	Settling time
M_p	Maximum overshoot
ess	Steady-stade error
GA	Genetic algorithm
DC	Direct current
ts_d	Settling time desired
M_{pd}	Maximum overshoot desired
ess_d	Steady-stade error desired
GM	Genetic model
v(t)	Voltage input of the motor
ia(t)	Current consumed by the motor
ω (t)	The angular speed of the motor
Ra	Armature resistance
La	Armature self-inductance
TL	Load torque
Km	Mechanical constant
Ke	Electrical constant
B	Friction coefficient
K_p	Proportional gain
K_i	Proportional gain
K_d	Proportional gain
K	Vector with three K gains [K_p K_i K_d]
RMSE	Root media square error
Logsig	Log-sigmoid transfer function
Purelin	Linear transfer function
Tansig	Hyperbolic tangent sigmoid transfer function

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

I. INTRODUCTION

PIDs are the most widely used controllers commercially due to their efficiency/simplicity ratio [1]. Its flexibility allows to find them in multiple areas. For example, at work [2], a PID is used to control a mobile firefighter robot. The authors of [3] use a PID controller to control the thermal environment variables in a pig farm, obtaining better performance than overhead electrical resistance with a thermostat. At work [4], the authors develop a two-degree PID to control the motor speed in an automatic lawnmower that uses solar energy as the primary source. Papers [5], [6] focus on motor speed control, demonstrating the usefulness of adaptive PIDs, in addition to comparing them with traditional PIDs; works like [7], [8] perform the same task but with a fuzzy PID presenting satisfactory results in regulating motor speed. These studies demonstrate the wide range of uses and the importance of using these types of controllers.

For the PID to generate the desired response in the system, its gains must be adjusted appropriately. The Ziegler Nichols standard method is the typical adjustment method. However, several studies indicate better results in tuning. For example, the work [9] uses fuzzy logic to auto-tune a PID, allowing the controller to vary according to the current system error.

Another alternative for tuning is the so-called metaheuristic algorithms. For example, the authors of [10]–[12] use a particle filter to perform the PID tuning. The works [13]–[15] use a similar methodology, in which GAs are used for the same objective, achieving effective results for the proposed tasks. Another metaheuristic algorithm used as a tuner is the cuckoo search algorithm, which has obtained good results as a tuner in works such as [16]–[18]. The use of metaheuristic algorithms has spread. In several investigations, such as [19]–[21], the authors analyze the advantages and disadvantages of these algorithms applied to engineering.

Despite the high adaptability and efficiency of metaheuristic algorithms, their process is inherently iterative [22]–[24]. This quality increases its execution time for online implementation. Another option highly used for PID auto-tuning is ANNs. Many works address this approach, like [22]–[24]. The investigation [25] presented a similar methodology to this work. Nevertheless, three fundamental characteristics of the dynamic response (Mp, ts, ess) for the ANN training are not considered.

The studies above indicate that PID adjustment is not a trivial issue. The present work proposes a PID self-tuning system based on ANNs to control DC motor speed. Unlike current tuning studies, the desired dynamic response is chosen through 3 parameters that reflect both the transitory stage and the steady-state. The backpropagation ANN trained with the database obtained by the GA is used to achieve this. The GA ensures data distribution in possible operating conditions and rejecting combinations of gains that make the system oscillate or make it unstable. The GA performs this work looking for the appropriate K values that would deliver the desired dynamic response.

The ANN uses as input the output of the motor and the step reference. The input vector representing the dynamic response is [M_p; ts; ess; reference]. On the other hand, the output vector is considered as [K_p K_i K_d]. These vectors allow the ANN to self-adjust the gains to control a DC motor under various operating conditions and references.

The methodology proposed in this work differs from other works on auto-tuning by ANNs like [22], [24], [26] in data distribution is not generally considered, nor the instability that various combinations of gains produce, nor the operating range of the system.

The dynamic response of the system can be different depending on the system under analysis and the control signal used. Figure 1 (a) shows the key components that determine the dynamic response of a system. The purpose of this work is to obtain a self-tuning procedure that allows choosing the desired response since, depending on the application, a different dynamic response may be required. Figure 1 (b) shows the possible outputs that the PID controller can have according to the combination of K chosen.

Some combinations generate an unstable response, and it is always desired to avoid this type of response. Other combinations generate a stable output but with high oscillations. Although it may be an acceptable response in a general way, it is desired to avoid oscillations. One useful combination generates a response without oscillations but with overshoot. This type of response is helpful when speed is required, and precision is not a critical factor. An example is surveillance with drones, where the vehicle is needed to move fast and overshoot it does not have significant effects on the mission. On the other hand, it is possible to obtain a slow but precise response as an output the system takes longer to reach the reference but does not have overshoot. For example, welding or a cutter robot cannot afford to go beyond the reference since an overshoot in these applications implies task collapses or severe failure. Considering the above, the system proposed in this article is presented as a self-tuning option that adapts to the response that the application requires and can make it faster or slower depending on the needs of the user or application.

The characteristics and contributions of this work can be summarized as:

- Construction of a database with GA for offline training of a backpropagation ANN.
- Profit combinations that make the system unstable thanks to GA are avoided in the database.
- The gains that make the system oscillate are also avoided.
- Whit the GA, data distribution in the database for network training is guaranteed, having data in all operable ranges of the physical system.

Selection of the desired dynamic response to find the appropriate gains with the auto-tuning system.

This work is divided into five sections. Section I presents the state-of-the-art, related investigations and a brief description of the proposed method. The database construction from

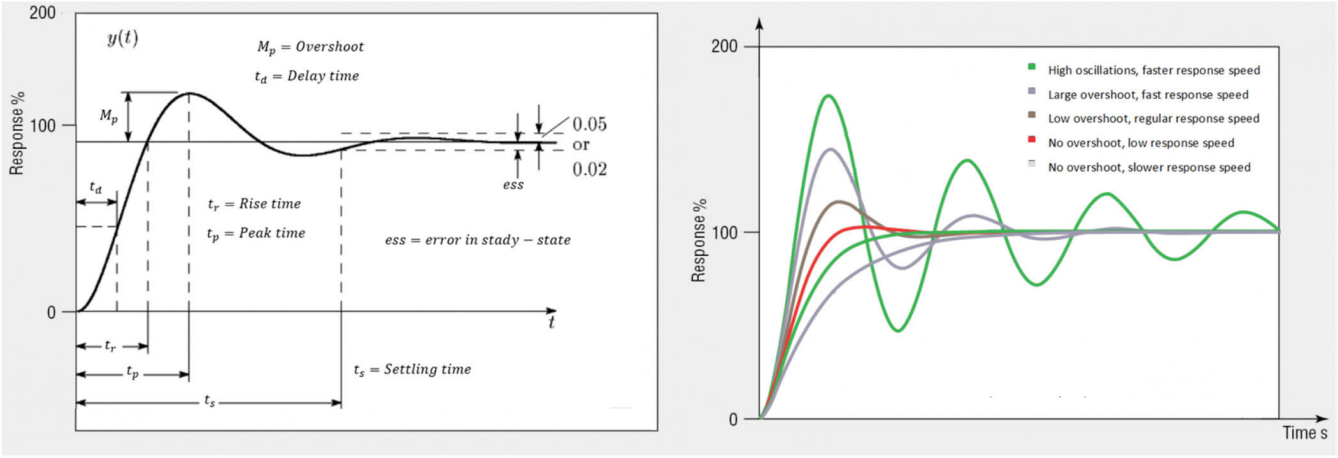

FIGURE 1. Dynamic response. (a) Parts of dynamic response (b) Types of dynamic response with different K.

a GA for controlling the speed of a DC motor was exhibiting in Section II. Section III describes the building and training of the ANN for the self-tuning of the PID. Section IV gives the results of the self-tuning PID. Finally, a conclusion is manifested in Section V.

II. DATABASE WITH GENETIC ALGORITHMS

GAs are inspired by the way genetics selects the best genes. They tested in multiple areas and have great flexibility in solving optimization problems [27]. GA was used because it is the most widespread and tested metaheuristic algorithm [27]. The objective of the GA is to build offline a database that relates the PID gains to their dynamic response. In that way, the algorithm looks for the gains that obtain a specific desired dynamic response. This work uses a fixed-state GA, in which the algorithm starts from a random population, where each individual is a vector with the three PID gains (K_p, K_i, and K_d). Taking into account the law of control of a PID described by (1).

$$U(t) = K_p e(t) + K_i \int e(t)\,dt + K_d \frac{de(t)}{dt} \tag{1}$$

The first step of the algorithm is the random creation of individuals. A large number of individuals provides large diversity. However, the computation time also increases with this number. For this work, the number of individuals (m) is 100. Later the fitness of each individual is evaluated to select the individuals with better fitness for this work. Fitness is assessed with the Euclidean distance between the ts, the M_p, and the ess, as shown in (2):

$$f = \sqrt{(ts_d - ts)^2 + (M_{pd} - M_p)^2 + (ess_d - ess)^2} \tag{2}$$

The f is the value of the fitness, and subscripts d means that it is the desired value. In this way, the individuals with a lower value of f will reproduce. It was also considered that the desired values are the maximum values expected in the response of the system. Thus, if the real value is less than the desired one, the contribution of that variable to the value of f will be zero. Later, it goes through the other GA stages, reproduction, mutation, and mixing of the descendants with the original surviving population. Figure 2 describes the whole process.

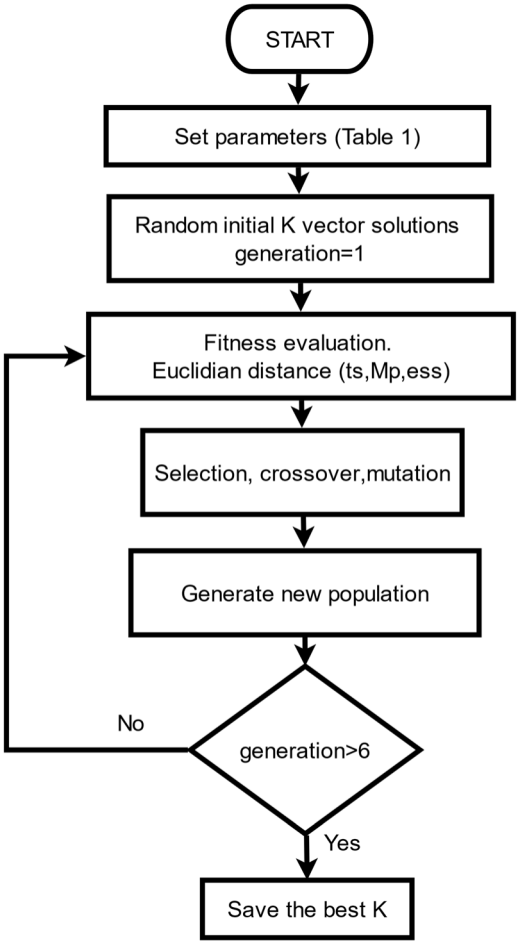

FIGURE 2. Block diagram for genetic algorithm proposed.

The parameters of the algorithm determine its efficiency in the search. The initial parameter values were chosen accord-

ing to the bibliography [27] and were adjusted according to the performance shown. Any solution found by the algorithm is valid, even if it is not the optimal solution. This allows choosing parameters with a certain tolerance. The complete list of the parameters used and their description are displayed in Table 1.

TABLE 1. Genetic algorithm parameters.

Parameter	Value	Description
Population size	100	Quantity of initial vector K gains
Generations	6	Number of iterations in which the algorithm will search
Range	[100 50 50]	The maximum value of each K [K_p K_i K_d]
Mutation probability	20%	The probability that a selected gene will undergo a mutation
Biological pressure	30%	Percentage of genes that will reproduce
Model	[ts_d M_{pd} ess_d Ref]	The vector that will use as a GM to follow
Individual	[K_p K_i K_d]	Vector with K random gains
Selection method	Rank Selection	The individual with the best fitness gets rank N, and the worst individual gets rank 1. The selection probability is $p(i) = \frac{rank(i)}{nx(n-1)}$
Crossover type	Single Point Crossover	A random point is selected for swapping chromosomes

The parameter values: population size, generations, mutation probability, biological pressure; were selected experimentally, starting from values used in previous works [26] and adjusting the value according to the algorithm performance. The range parameter (limit values of K gains) by definition must be greater than zero. Hence it is only necessary to set an upper limit. The upper limit for each K was made considering the saturation of the plant effect and dynamic response. Using the values described in Table 1 is possible to find gains that generate the desired response in different ranges. The plant used in this work is a DC motor model CML-050 from the Mavilor brand. DC motors are widely studied plants. The system of differential equations described in (3) represents the dynamic model of a plant.

$$\left.\begin{aligned} v(t) &= R_a i_a(t) + L_a \frac{di_a(t)}{dt} + Ka_a \omega(t) \\ K_m i_a(t) &= J\frac{d\omega(t)}{dt} + B\omega(t) + T_L \end{aligned}\right\} \tag{3}$$

where:

v(t) is the voltage.

ia(t) is the current.

ω (t) is the angular velocity.

Additionally, Table 2 indicates the model parameters of the chosen motor. These parameters were obtained following the work exhibited in [28], where the authors use the Steiglitz-McBride algorithm for the parametric estimation. The data acquisition system consists of two self-developed cards. The first card is used to condition the signal. Later, the data acquisition card based on the PIC18F455 microcontroller is used. This card processes the signals (current and speed) and sends them via USB to the computer, where Matlab uses the Steiglitz-McBride algorithm and obtains the motor parameters. The general schematic process for parameter estimation is illustrated in Fig. 3.

TABLE 2. Mavilor CML-050 parameters.

Parameter	Value	Description
R_a	3.136309 Ω	Armature resistance
L_a	0.013070 H	Armature self-inductance
T_L	0 Nm	Load torque
K_m	0.048774	Mechanical constant
K_e	0.048774	Electrical constant
B	$0.000169 \frac{Kg.m^2}{s^2}$	Friction coefficient
J	0.000009 N. m	Moment of inertia
Nominal voltage	24 V	Nominal operating voltage
Nominal velocity	314 rad/s	Nominal operating velocity

It is required to clear the variables of interest to simulate the dynamic motor model. Since it is speed control, the system of equations (3) is rewritten, such as:

$$\left.\begin{aligned} \frac{di_a(t)}{dt} &= \frac{v(t) - R_a i_a(t) + Ka_a \omega(t)}{La} \\ \frac{d\omega(t)}{dt} &= \frac{K_m i_a(t) - B\omega(t) - T_L}{J} \end{aligned}\right\} \tag{4}$$

Once the plant and the necessary parameters for the algorithm have been defined, the GA runs iteratively. The process is displayed in Fig. 4, where the plant (the Mavilor motor) is simulated by (4). This process is repeated with random values for the genetic model (GM) to evaluate cases with different combinations of ts_d, M_{pd}, ess_d, and the reference.

Subsequently, Gains that meet the GM requirements are save in the database or, failing that, the closest solution. These values are stored in the database that will be used for neural training in the next stage. Table 3 shows the ranges used to generate the other GM combinations. A total of 10296 combinations of different GMs were simulated for the construction of the database. The values were selected to keep the response within the operating limits of the motor.

TABLE 3. Values used for database generation.

Parameter	Range
ts	[0.05, 2.25]
M_p	[0 10]
ess	[0 5]
ref	[5 315]

The database used has vectors in which ts, Mp, ess, reference, Kp, Ki, and Kd are stored. The final database is 10,296 vectors with simulation outputs of the proposed GM

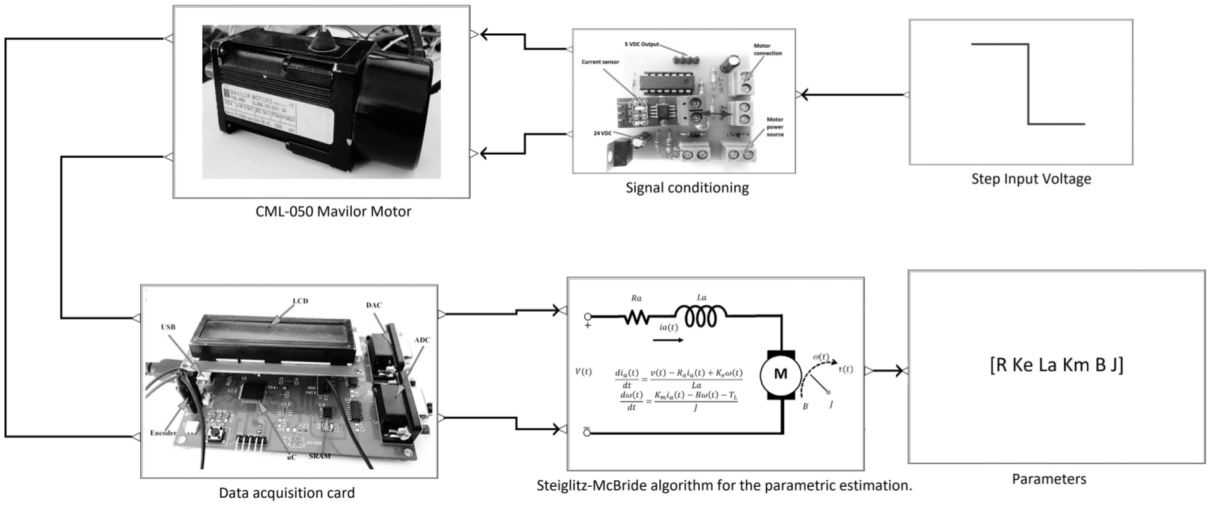

FIGURE 3. Block diagram for parameter estimation.

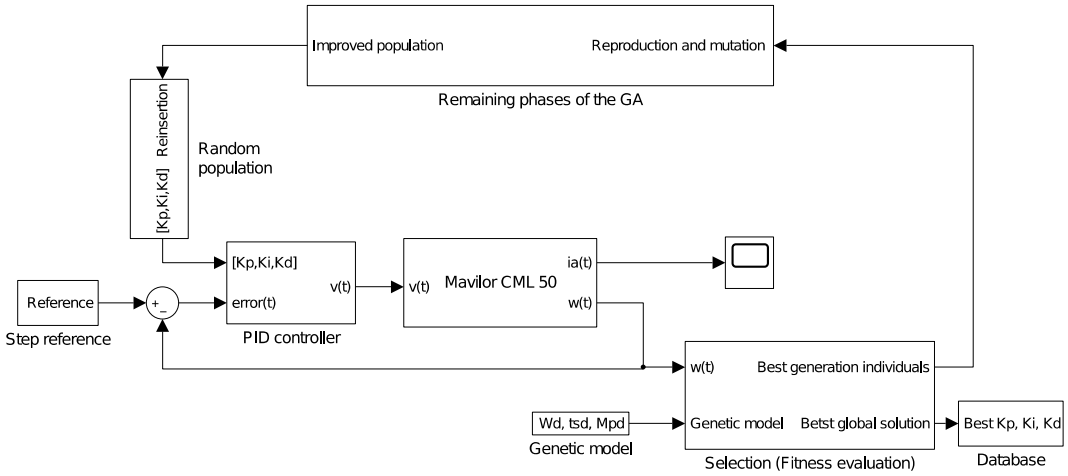

FIGURE 4. Block diagram of the procedure for obtaining the database with the genetic algorithm.

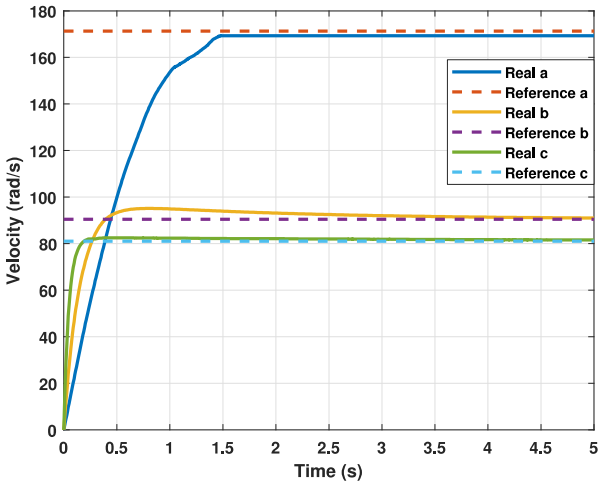

FIGURE 5. Genetic algorithm results: Motor velocity versus reference.

were stored. Several results of this algorithm can be seen graphically in Fig. 5 and numerically in Table 4.

The complete database has a root media square error (RMSE) of 5%. It meets the requirements of the GM

TABLE 4. Genetic algorithm numeric results.

Test	Genetic model [ts_d, M_{pd}, ess_d, Ref]	Gains [K_p, K_i, K_d]	Results [ts M_p ess]
a	[1.4, 0, 2.5, 171]	[6.3, 5.09, 4.83]	[1.36, 0, 2.03]
b	[2.3, 6, 1, 90]	[31.7, 16.2, 4.5]	[2.2189 ,5.1, 0.53]
c	[0.2, 2, 1, 81]	[100, 24, 4.7]	[0.16, 1.74, 0.52]

requested, keeping the values below the maximum desired values for most of the cases. In the cases where the conditions were not met, the GA obtained values close to the model. Therefore, these solutions are kept in the database.

III. ARTIFICIAL NEURAL NETWORK AS AUTOTUNER

A backpropagation ANN is developed in this segment. This type of ANN was selected because it is the most widely used neural network model [29]. The ANN was trained with the database obtained in the previous section. The database was divided into inputs (ts, Mp, ess, Ref) and outputs (Kp, Ki, Kd) to work with ANN. In turn, each submatrix was randomly divided as follows: 70% of the data for the training stage,

15% for a network validation stage, and 15% for evaluating the network performance.

The first stage in ANNs consists of selecting the appropriate parameters for the architecture. In this work, the initial architecture is based on the geometric pyramid rule. Subsequently, it adapted following the methodology exhibited in [30], where the authors adjust the architecture according to the network performance. Multiple simulations were run to test performance with various architectures, taking RMSE as a performance measure. The most relevant results of these tests can be seen in Table 5.

TABLE 5. Performance of neural networks with different architectures.

Net	Hidden layers	Activation function	Train	Validation	Test
			RMSE	RMSE	RMSE
Net1	6	tansig- logsig	22.76%	23.21%	22.92%
Net2	12	tansig - logsig	22.51%	22.64%	23.14%
Net3	12	tansig - purelin	15.72%	15.78%	15.55%
Net4	12	tansig - tansig	15.67%	15.43%	15.71%
Net5	8-4	tansig - tansig - purelin	15.61%	15.75%	15.75%
Net6	14-8	purelin - tansig - purelin	14.99%	15.36%	15.46%
Net7	14-8	tansig - tansig - tansig	14.89%	14.90%	15.60%
Net8	18-9	tansig - tansig - purelin	15.02%	15.54%	15.15%
Net9	**18-9**	**tansig - tansig - tansig**	**14.94%**	**15.44%**	**14.83 %**
Net10	34-20	tansig - tansig - tansig	14.73%	15.44%	15.76%

In Table 5, it is observed that architecture 9 is the network that offers the best output, presenting a lower RMSE. This error is the average RMSE of each gain (K_p, K_i, K_d). Thus, this architecture was used. As an alternative, a network for each K (without dependence between outputs) was developed. However, the error is similar. Therefore, it was decided to work with a single network for the vector K.

It is necessary to analyze each variable influence on the network. Some typical dynamic response variables were not considered, such as rise time, delay-time, and peak-time, because these variables have a high correlation with the overshoot. Having a multivariable system is challenging to evaluate the impact of one of these variables in the network. Nevertheless, to verify that the inputs are significant for the ANN, the same architecture was executed but eliminating the input variable from the network inputs. The results can be seen in Table 6.

Contemplating Table 6, it is noticed that each variable contributes a significant reduction of the RMSE. Therefore, the entries selected for the network can be considered valid. In Fig. 6 can be observed the general process of the proposed autotuner with this network. It can be seen that the network requires all four parameters of the GM. With this data, the trained network will calculate the three appropriate

TABLE 6. Influence of each input variable in the neural network.

Variable removed	Train RMSE	Validation RMSE	Test RMSE
ts	17.41%	17.16%	17.84%
M_p	18.27%	18.47%	18.61%
ess	17.01%	17.23%	17.41%
Ref	16.39%	16.92%	17.11%

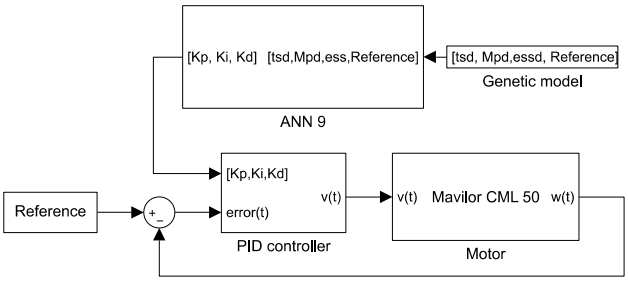

FIGURE 6. Block diagram of the proposed automatic adjustment system in simulation.

gains. Subsequently, the PID can obtain the desired dynamic response.

The Matlab-Simulink environment was used to execute the simulations. Matlab runs the GM while Simulink is in charge of simulating the PID, the ANN, and the motor dynamics. For Simulink simulations, a variable step numeric method (ODE45) was used for 5 s.

IV. RESULTS

This section shows the ANN results as a tuner agent, which is considered desired establishment times, desired maximum impulses, desired maximum steady-state errors, and desired speeds that the network has not used in its training. The result of the test data set is 14.83% in RMSE. This number allows us to estimate the output globally. Nevertheless, this RMSE is in gains estimation. Ten random GM combinations were proposed to evaluate the performance of the ANN in the dynamic response. These values combination was not previously used in any data set. The selected values are displayed in Table 7.

TABLE 7. New genetic models are used to test the network.

Model	Reference (rads/s)	ts_d (s)	M_{pd} (%)	ess_d (%)
GM1	150.5	0.51	4.25	4.8119
GM2	127.1	0.193	7.43	8.023
GM3	202.0	0.0341	9.95	2.6316
GM4	50.2	1.874	8.97	0.6192
GM5	197.5	0.165	9.61	4.62
GM6	38.5	0.414	6.87	5.9112
GM7	64.1	2.101	0.01	9.780
GM8	221.9	0.094	3.22	2.4274
GM9	88.6	1.201	4.17	2.9104
GM10	90.8	1.810	0.54	1.6365

The values shown in Table 7 were tested under the same conditions, that is, a Simulink simulation with ODE45 for 5 s. In the same way, initial conditions were considered as zero for both velocity and current. The non-linearity of saturation was also used in work with the expected operating ranges in the motor (0 to 24 v). In this way, the current is also limited.

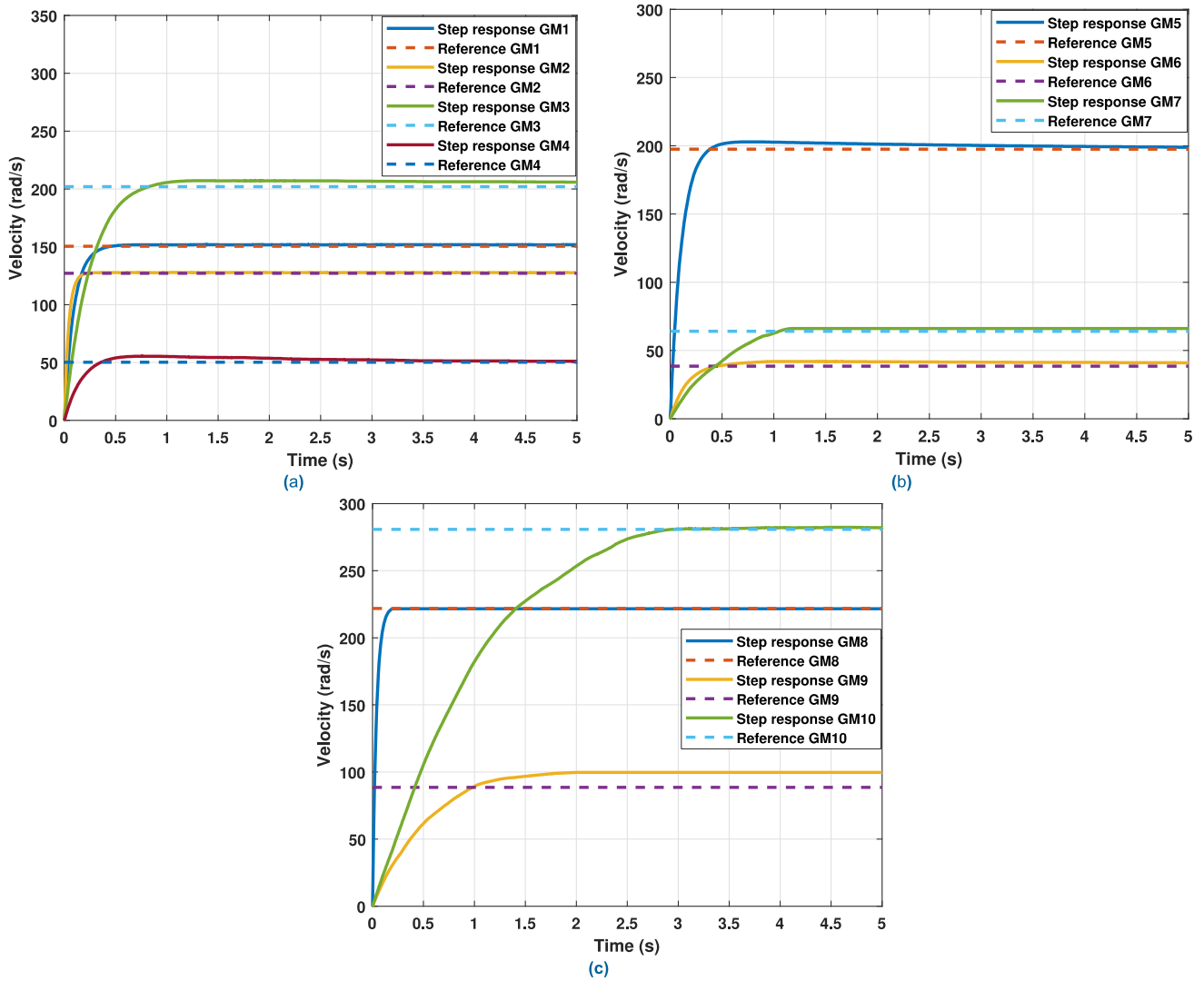

FIGURE 7. ANN simulation response with K_p, K_i, K_d, estimated. (a) Velocity response from GM1 to GM4. (b) Velocity response from GM5 to GM7. (c) Velocity response from GM8 to GM10.

TABLE 8. Numerical network results for each genetic model.

Model	K_p	K_i	K_d	ts(s)	M_p(%)	ess(%)
GM1	68.98	0.02	6.51	0.29	0.69	0.82
GM2	30.39	0.01	1.29	0.12	0.40	0.44
GM3	98.48	16.63	25.33	0.66	2.27	1.84
GM4	68.75	36.57	10.75	2.03	9.33	1.67
GM5	75.25	23.65	8.26	0.29	2.45	0.68
GM6	75.41	7.88	11.92	0.48	8.09	6.04
GM7	99.67	0.01	48.86	1.01	2.92	3.03
GM8	63.79	12.72	2.51	0.11	0.31	0.16
GM9	99.78	0.01	48.38	1.26	11.00	11.14
GM10	24.95	6.61	31.91	2.35	0.36	0.40

The results are presented in Table 8 are those obtained using the GMs from Table 7. The GMs represent different operating conditions that a user may need according to the application required.

Additionally, Fig. 7 illustrates the step response of the GAs proposed. The dynamic response with K_p, K_i, and K_d estimated by the ANN can be observed. The previous images allow us to monitor the performance for specific cases. The overall network performance is evaluated from the dynamic response point of view with a new data set. The size of the new data is the same as the original. The errors for each input variable are shown in Table 9.

TABLE 9. Error in step response for each variable.

	ts	M_p	ess
RMSE	18.61%	15.78%	09.19%

Also, tests were performed to observe an input parameter variation while the other parameters remain fixed. The varied parameter will cover the complete ranges of motor operation. In this way, it is possible to observe the network performance in different magnitudes of a parameter. With parameter variation, further gains were calculated with the ANN, and the step response given by the system was analyzed. For these tests, the GM9 in Table 8 was used, which had the worst results in the study for specific models. The first input to vary is ts_d.

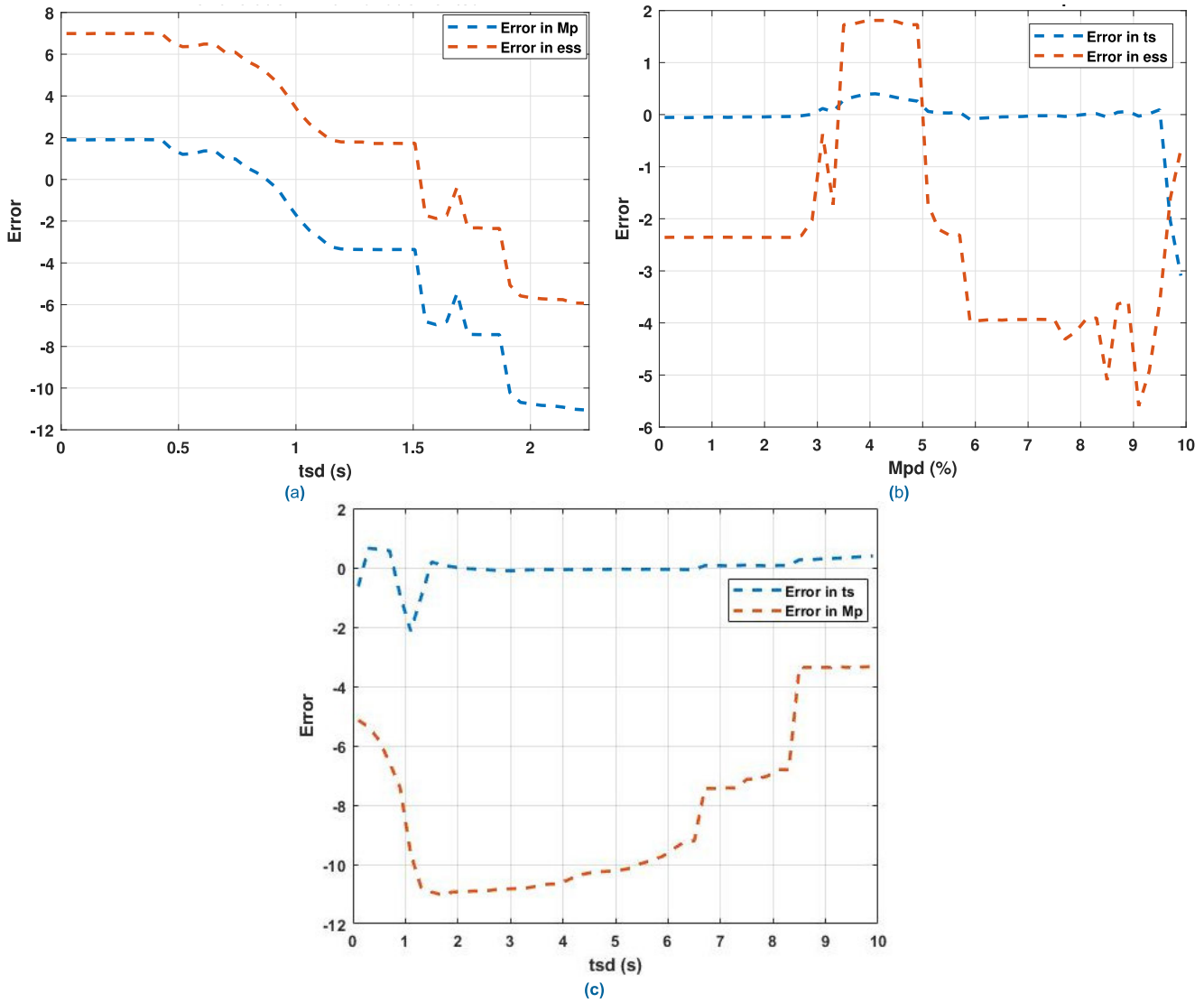

FIGURE 8. Error change with variation of a parameter. (a) Variation of ts_d. (b) Variation of M_{pd}. (c) Variation of ess.

Figure 8a exposes the error variation (desired value- real value) in the other inputs as the value of ts changes. Figure 8b illustrates the error variation when the value of M_{pd} changes. Finally, Fig. 8c displayed the error variation when the percentage of essd changes. Although there are variations, the above images indicate that the network performance varies according to the range in which the input parameters work.

Finally, Table 10 shows a comparison of the ANN-based autotuners. This work considers two fundamental aspects, the dynamic response and, due to the GA, it avoids combinations that produce instability.

It can be observed in Table 10 that the difference of this work concerning similar titles lies in managing the dynamic response. Typically, only the error is supervised; however, controlling how the error is reduced is critical for many processes.

The results obtained with the test data set used to test the performance of the proposed autotuner show that 86% of the data (1327 combinations) meet the desired requirements or a lower value, that is: ts <tsd, ess <essd and Mp <Mpd. The 14% that do not meet a lower value have an approximate value, as GM9 in Table 8. Also, 100% of the test data are shown to provide a stable response. Although it does not guarantee the stability for an infinite set of possible combinations, the results suggest stable responses in the operation ranges,

TABLE 10. Comparison of ANN-based autotuner in similar works.

Work	Variables considered						
	Reference	ts	ess	M_p	error	Instability	PID Output
Our Work	x	x	x	x	x	x	-
[22]	x	-	-	-	x	-	x
[23]	x	-	-	-	x	-	-
[24]	x	-	-	-	x	-	x
[25]	x	-	-	-	x	-	-
[26]	-	-	-	-	e_t e_{t-1} e_{t-2}	-	-

acceptable error reduction performance, and, above all, control of the dynamic response, which is the main objective of the work.

V. CONCLUSION

This article presents an auto-tuning system using ANNs that allows controlling the speed of a DC motor and the type of response expected. The results indicate that auto-tuning can follow references in multiple plant operating ranges and maintain the desired temporal response and avoid the gains combinations that can make the system unstable. The dynamic response of the system complies with the desired GA even when operating conditions are required at the motor running limits, for example, ts of 0.1 s or speeds close to the limit of 314 rad / s with nominal operating voltage.

Unlike other auto-tuning investigations with ANNs, in this work, a database built with GA was used. The GA guarantees that the training data of the network do not include gains combinations that cause oscillations. Since they comply with the design requirements proposed, and this let us choose the desired dynamic response.

Although the integral of the error is typically used as a fitness function, this work focuses on the response of the system. Therefore, characteristics of the dynamic response are used. The first characteristic used is overshoot, which occurs in the transient stage. The second characteristic used is the steady-state error, which appears in the steady-state. Finally, the settling time was used, representing the transition between the two dynamic response stages. In this way, the fundamental parts were covered in the output system with the chosen fitness function.

For the development of this work, some simplifications and limitations were considered. For example, the architecture of the neural network is limited to backpropagation networks. Regarding the motor model, it is assumed that Ka and Ke are equal in magnitude, although in reality, their values differ slightly. The entire test dataset was tested with the ANN, and no instability was found in those data. However, stability cannot be guaranteed for the infinite set of input combinations. Finally, it is known that there is a strong correlation between the rise time, delay-time, and peak-time with the overshoot. For this reason, these variables of the dynamic response were not considered as network inputs.

A disadvantage of the proposed method is that the performance of the network is linked to the operating range of the systems, as shown in Fig. 8. this implies that although you have control over the response of the system, the precision will not be constant and in the system operating limits performance declines. Another limitation is that some combinations are physically impossible to achieve for the system. However, the proposed method does not consider any way to identify these combinations.

A study was done to determine the appropriate network architecture, testing the performance of different networks. The training and execution of the network did not represent a significant computation time. However, the GA simulated with the dynamic model of the motor if it has a high computational cost. It could be reduced with a simpler metaheuristic algorithm from the computational point of view, like the algorithm of Jaya or even algorithms that require fewer iterations to converge, like the cuckoo search algorithm. As future work, this methodology will optimize the energy consumption of a drone based on the task that the vehicle must perform.

ACKNOWLEDGMENT

The authors would like to acknowledge the support of Universidad Politécnica de Querétaro in the production of this work.

REFERENCES

[1] R. P. Borase, D. K. Maghade, S. Y. Sondkar, and S. N. Pawar, "A review of PID control, tuning methods and applications," *Int. J. Dyn. Control*, to be published, doi: 10.1007/s40435-020-00665-4.
[2] A. Latif, K. Shankar, and P. T. Nguyen, "Legged fire fighter robot movement using PID," *J. Robot. Control (JRC)*, vol. 1, no. 1, pp. 15–19, 2020, doi: 10.18196/jrc.1104.
[3] J. de Souza Granja Barros, L. A. Rossi, and Z. Menezes de Souza, "PID temperature controller in pig nursery: Spatial characterization of thermal environment," *Int. J. Biometeorol.*, vol. 62, no. 5, pp. 773–781, May 2018, doi: 10.1007/s00484-017-1479-x.
[4] M. R. Habib, K. Ahmed, N. Khan, M. R. Kiran, M. A. Habib, M. T. Hasan, and O. Farrok, "PID controller based automatic solar powerdriven grass cutting machine," in *Proc. Int. Conf. Comput., Commun., Chem., Mater. Electron. Eng. (IC4ME2)*, Jul. 2019, pp. 1–4, doi: 10.1109/IC4ME247184.2019.9036513.
[5] J.-W. Jung, V. Q. Leu, T. D. Do, E.-K. Kim, and H. H. Choi, "Adaptive PID speed control design for permanent magnet synchronous motor drives," *IEEE Trans. Power Electron.*, vol. 30, no. 2, pp. 900–908, Feb. 2015, doi: 10.1109/TPEL.2014.2311462.
[6] S. K. Suman and V. K. Giri, "Speed control of DC motor using optimization techniques based PID controller," in *Proc. IEEE Int. Conf. Eng. Technol. (ICETECH)*, Mar. 2016, pp. 581–587, doi: 10.1109/ICETECH.2016.7569318.
[7] Y. A. Almatheel and A. Abdelrahman, "Speed control of DC motor using fuzzy logic controller," in *Proc. Int. Conf. Commun., Control, Comput. Electron. Eng. (ICCCCEE)*, Jan. 2017, pp. 1–8, doi: 10.1109/ICCCCEE.2017.7867673.
[8] D. Somwanshi, M. Bundele, G. Kumar, and G. Parashar, "Comparison of fuzzy-PID and PID controller for speed control of DC motor using LabVIEW," *Procedia Comput. Sci.*, vol. 152, pp. 252–260, 2019, doi: 10.1016/j.procs.2019.05.019.
[9] H. Jigang, W. Jie, and F. Hui, "An anti-windup self-tuning fuzzy PID controller for speed control of brushless DC motor," *Automatika*, vol. 58, no. 3, pp. 321–335, Jul. 2017, doi: 10.1080/00051144.2018.1423724.
[10] A. Noordin, M. A. Mohd Basri, Z. Mohamed, and A. F. Zainal Abidin, "Modelling and PSO fine-tuned PID control of quadrotor UAV," *Int. J. Adv. Sci., Eng. Inf. Technol.*, vol. 7, no. 4, p. 1367, Aug. 2017, doi: 10.18517/ijaseit.7.4.3141.
[11] M. Ali, I. Umami, and H. Sopian, "Particle swarm optimization (PSO) Sebagai tuning PID Kontroler Untuk Kecepatan motor DC," *Jurnal Intake: Jurnal Penelitian Ilmu Teknik dan Terapan*, vol. 7, no. 1, pp. 10–20, 2016.
[12] A. T. Azar, H. H. Ammar, Z. F. Ibrahim, H. A. Ibrahim, N. A. Mohamed, and M. A. Taha, "Implementation of PID controller with PSO tuning for autonomous vehicle," in *Proc. Int. Conf. Adv. Intell. Syst. Inform.*, 2020, pp. 288–289, doi: 10.1007/978-3-030-31129-2_27.
[13] D. C. Meena and A. Devanshu, "Genetic algorithm tuned PID controller for process control," in *Proc. Int. Conf. Inventive Syst. Control (ICISC)*, Jan. 2017, pp. 1–6, doi: 10.1109/ICISC.2017.8068639.
[14] N. P. Putra, G. J. Maulany, F. X. Manggau, and P. Betaubun, "Attitude quadrotor control system with optimization of PID parameters based on fast genetic algorithm," *Int. J. Mech. Eng. Technol.*, vol. 10, no. 1, pp. 335–343, 2019.
[15] S. Tiwari, A. Bhatt, A. C. Unni, J. G. Singh, and W. Ongsakul, "Control of DC motor using genetic algorithm based PID controller," in *Proc. Int. Conf. Utility Exhib. Green Energy Sustain. Develop. (ICUE)*, Oct. 2018, pp. 1–6, doi: 10.23919/ICUE-GESD.2018.8635662.

[16] K. S. M. J. Singh, I. Elamvazuthi, K. Z. K. Shaari, and F. V. Lima, "PID tuning control strategy using cuckoo search algorithm," in *Proc. IEEE Student Conf. Res. Develop. (SCOReD)*, Dec. 2015, pp. 129–133, doi: 10.1109/SCORED.2015.7449309.

[17] K. S. M. Jagindar Singh, I. Elamvazuthi, K. Z. K. Shaari, and N. Perumal, "Development of PID controller tuning tool based on cuckoo search algorithms," in *Proc. IEEE 3rd Int. Symp. Robot. Manuf. Autom. (ROMA)*, Sep. 2017, pp. 1–5, doi: 10.1109/ROMA.2017.8231738.

[18] D. Fister, I. Fister, I. Fister, and R. Šafarič, "Parameter tuning of PID controller with reactive nature-inspired algorithms," *Robot. Auto. Syst.*, vol. 84, pp. 64–75, Oct. 2016, doi: 10.1016/j.robot.2016.07.005.

[19] A. Gogna and A. Tayal, "Metaheuristics: Review and application," *J. Exp. Theor. Artif. Intell.*, vol. 25, no. 4, pp. 503–526, Dec. 2013, doi: 10.1080/0952813X.2013.782347.

[20] X. S. Yang, "Review of meta-heuristics and generalised evolutionary walk algorithm," *Int. J. Bio-Inspired Comput.*, vol. 3, no. 2, pp. 77–84, 2011, doi: 10.1504/IJBIC.2011.039907.

[21] X. S. Yang and S. Deb, "Engineering optimisation by cuckoo search," *Int. J. Math. Model. Numer. Optim.*, vol. 1, no. 4, p. 330, 2010, doi: 10.1504/IJMMNO.2010.035430.

[22] R. Hernández-Alvarado, L. García-Valdovinos, T. Salgado-Jiménez, A. Gómez-Espinosa, and F. Fonseca-Navarro, "Neural network-based self-tuning PID control for underwater vehicles," *Sensors*, vol. 16, no. 9, p. 1429, Sep. 2016, doi: 10.3390/s16091429.

[23] A. Zribi, M. Chtourou, and M. Djemel, "A new PID neural network controller design for nonlinear processes," *J. Circuits, Syst. Comput.*, vol. 27, no. 04, Apr. 2018, Art. no. 1850065, doi: 10.1142/S0218126618500652.

[24] S. Bari, S. S. Zehra Hamdani, H. U. Khan, M. U. Rehman, and H. Khan, "Artificial neural network based self-tuned PID controller for flight control of quadcopter," in *Proc. Int. Conf. Eng. Emerg. Technol. (ICEET)*, Feb. 2019, pp. 1–5, doi: 10.1109/CEET1.2019.8711864.

[25] P. A. Chertovskikh, A. V. Seredkin, O. A. Gobyzov, A. S. Styuf, M. G. Pashkevich, and M. P. Tokarev, "An adaptive PID controller with an online auto-tuning by a pretrained neural network," in *Proc. J. Phys., Conf.*, 2019, vol. 1359, no. 1, doi: 10.1088/1742-6596/1359/1/012090.

[26] T. Jiménez, N. Merayo, and R. J. Durán, "Adaptive-tuning method based on neural networks for PID controllers applied to passive optical networks (PONs)," in *Proc. 17th Int. Conf. Transparent Opt. Netw. (ICTON)*, Jul. 2015, pp. 1–4, doi: 10.1109/ICTON.2015.7193425.

[27] S. Mirjalili, J. Song Dong, A. S. Sadiq, and H. Faris, "Genetic algorithm: Theory, literature review, and application in image reconstruction," in *Studies in Computational Intelligence*. Cham, Switzerland: Springer, 2020.

[28] H. Paredes, J. Miguel, M. B. Benigno, and R. A. Omar, "Sistema de identificación paramétrica para motores de corriente directa," *La Mecatrónica en México*, vol. 8, no. 3, pp. 115–130, 2019.

[29] H. Mustafidah and Suwarsito, "Correlation analysis between error rate of output and learning rate in backpropagation network," *Adv. Sci. Lett.*, vol. 24, no. 12, pp. 9182–9185, Dec. 2018, doi: 10.1166/asl.2018.12121.

[30] M. Vakili, S. R. Sabbagh-Yazdi, S. Khosrojerdi, and K. Kalhor, "Evaluating the effect of particulate matter pollution on estimation of daily global solar radiation using artificial neural network modeling based on meteorological data," *J. Cleaner Prod.*, vol. 141, pp. 1275–1285, Jan. 2017, doi: 10.1016/j.jclepro.2016.09.145.

OMAR RODRÍGUEZ-ABREO received the master's degree in mechatronics from the University of Malaga and the Ph.D. degree from Universidad Iexpro. He is currently pursuing the Ph.D. degree in advanced technology with the Instituto Politécnico Nacional (IPN). He has worked as a Development Engineer with Indesyth, where he developed industrial prototypes funded by the Mexican National Council on Science and Technology (CONACYT). He is currently a Professor in manufacturing technology engineering educational program with Universidad Politécnica de Querétaro (UPQ). He is also a Mechatronics Engineer. He also works as a Researcher in control, mobile robotics, fuzzy logic, and metaheuristic algorithms.

JUVENAL RODRÍGUEZ-RESÉNDIZ (Senior Member, IEEE) was with West Virginia University, as a Visiting Professor, in 2012. He is currently the coordinator for the master's degree in automation with Querétaro State University (UAQ), México. He is also the Director of the Office for Partnership with Industry and Academy, UAQ. Herein, he has taught more than 150 digital signal processing and research methodology courses. He also belongs to the Mexican Academy of Sciences, the Mexican Association of Robotics and Mechatronics, and the National Research Academy, México. He has developed more than 40 industrial projects by linking UAQ and government. His team has published more than 100 technical and education articles. He patented more than ten innovations. He has won several national and international prizes for his academic and innovation developments. He has been the advisor of more than 200 theses of undergraduate, master, and doctoral grades. He has been invited to give 30 conferences around the world.

CARLOS FUENTES-SILVA received the M.S. and Ph.D. degrees in intelligent control from the Universidad Autónoma de Querétaro (UAQ), Mexico. He is currently a Professor in manufacturing technology engineering educational program with Universidad Politécnica de Querétaro (UPQ), Mexico. His main research interests include fuzzy decision-making, neural networks, machine learning, and computer vision. He is also working on autonomous guided vehicles with computer vision for road, pedestrians, and traffic signals detection, and generating navigation reports to a digital dashboard over the Internet of Things (IoT).

RODRIGO HERNÁNDEZ-ALVARADO received the Ph.D. degree in mechatronics from the Center for Engineering and Industrial Development (CIDESI), México, in 2016. He is currently a Professor with the Division of Industrial Technology, Universidad Politécnica de Querétaro. His research interests include control and implementation, intelligent controls oriented to tracking, mobile, and underwater robotics.

MARÍA DEL CONSUELO PATRICIA TORRES FALCÓN received the degree in actuarial studies and the master's degree in transportation engineering from UNAM and the Ph.D. degree in advanced technology from IPN. In 1994, she addressed creating a model to generate fuel efficiency inland cargo units at the PEMEX-Gas Institution. In 1997, she collaborated as a Consultant of the Binational Project "Border Transportation Mexico-United States" with the Consultant Group Partners International, Government of Arizona, USA. She has the intellectual property *SIMULATOR 2020* and in the process of a utility model of an Arduino embedded system to control diesel consumption of transport units terrestrial cargo (UTTC) in real-time, in 2020. She has collaborated in the analysis of liquefied gas at the national level. She has the desirable PRODEP profile. Since 2017, she has been participating as an Evaluator of the scientific and technological projects with the Polytechnic University of Querétaro. She has a pending publication of the book *Compendio de Forecasts e Inventario* (2020).