Real-time optimized wireless networked control system with cooperative network protocols

Yousif Safaa Alobaidy, Osama Ali Awad
Department of System Engineering, College of Information Engineering, Al-Nahrain University, Baghdad, Iraq

Article Info

Article history:

Received Dec 28, 2022
Revised May 1, 2023
Accepted May 6, 2023

Keywords:

ESP8266
Fuzzy PID controller
NodeMCU
TCP/IP
TrueTime simulator
User datagram protocol
Wireless networked control system

ABSTRACT

In this paper, we present a real-time optimized fuzzy fuzzy proportional integral derivative (FPID)-controlled wireless networked system for a high-torque direct current (DC) motor. The main challenge faced by such systems is the delay in the wireless networked control system (WNCS). We employed a powerful FPID controller tuned using particle swarm optimization (PSO) technique to compensate for the delay. The system is tested on a network using the TrueTime simulator with different parameters. The results show that the system exhibits a very stable response, with the FPID controller compensating for the delay effectively. Increasing the number of nodes negatively impacts the system's performance, resulting in higher overshoot, longer settling time, and longer rise time. Moreover, the choice of bandwidth share and sampling time significantly affects the system's stability and real-time response. The use of transmission control protocol/internet protocol (TCP/IP) or user datagram protocol (UDP) protocols with Node MCU is necessary to transfer data from the Arduino Microcontroller to MATLAB, as MATLAB TrueTime simulator does not support direct serial communication. In conclusion, this study highlights valuable insights into the performance of the proposed system, demonstrating the need for further improvements in the system's design and control algorithms to achieve stable operation.

Corresponding Author:

Yousif Safaa Alobaidy
Department of System Engineering, College of Information Technology, Al-Nahrain University
Baghdad, Iraq
Email: yousif_saffa97@yahoo.com

1. INTRODUCTION

A networked control system is made up of interconnected control loops that are accessed through communication networks. The controller and the feedback signals are exchanged between the system and the controller in this type of control scheme [1]. A wireless networked control system (WNCS) is the same case except for physical connectivity between the networks meaning that the controllers, actuators, and sensors that compose the network communicate through wireless signals instead of the conventional point-to-point wired connection [1]. The use of wireless networks has increased multiple times in recent years with advancements in the fields of wireless communication and the introduction of 4th and 5th-generation communication systems, mostly due to increased facility regarding maintenance control, extended flexibility, and simpler installation [2]. WNCS is introduced to remove networked control system (NCS) issues such as retardant in ease of use and portability because of the wired connectivity. The network delay will result in loss of packets thus resulting in degrading system performance and stability. Despite the obvious advantages of wireless technology, the problem that remains is the time delay in the feedback loop between the controller and sensor/actuator nodes which result in the corresponding packet loss combined with the network capacity limitation in the network, which decrease the system performance and lead to system instability.

The researchers presented different suggestions regarding different plans to handle real-time wireless network control. Lee and Yoo [3] implemented an industrial wireless network to control a real-time system operating in an industrial environment. An industrial network in which EtherCAT wired technology and IEEE 802.15.4e mixed wireless technology is used to evaluate this approach. A structure of the inverted pendulum system is used as a reference model for real-time systems for performance evaluation. Through the use of an embodied industrial system for performance evaluation of the wireless network, it is confirmed that it can be applied to real-time systems operating under strict requirements.

Lu *et al.* [4] stated that recently, there have been a number of advances in real-time wireless sensor and actuator networks (WSANs). These advancements will be used in industrial control systems, scheduling algorithms, and analyses. In addition to this usage, WSANs will also be tested for their ability to cyber-physical co-design with wireless controllers. Finally, the development process for these types of systems will be evaluated using a wireless cyber-physical simulator. Razfar *et al.* [5] designed a network to be used in rocket delivery launch vehicles in real-time considering the delay of the network from the environment and the noise thus proposing a system that gathers acceleration information from multiple sensors relying on the orientation of the rocket back the system. These factors analysis was done using an OPNET simulator to understand the sensor's performance.

Liao *et al.* [6], a proposal was made to create a remote-control infrared system using the internet. The proposition combined both computer and network technology. First, commands were decoded from an input given through the use of web and network technologies. An intelligent self-learning system was built that had a remote-control capability using infrared signals and linkages between a personal computer (PC) and web server in order to provide summary information about the PC's ability to remotely access home appliances via the internet. Liu *et al.* [7] one of the proposed improvements to proportional integral derivative (PID) controllers is updating their output scaling factor on a continual basis. In order for this proposal to be effective, a fusion-weighted summation rule should be used in parallel with other controller types. The simulation results showed that using this approach improved performance compared to traditional methods. Maity *et al.* [8] a type-2 fuzzy logic controller was used to control a direct current (DC) servo motor. The fuzzy logic controller (FLC) has the advantage of being able to withstand disturbances and parameter uncertainties, but it is computationally more complex than a type-1 fuzzy PID controller. The performance of the controller in the presence of noise and load conditions on the DC servo motor application was satisfactory with respect to an interval IT2-POFPID.

Dey *et al.* [9] proposed fuzzy PID controller was found to have improved performance when compared to other controllers. The self-tuning ability allowed for continual updating of output scaling factors, which led to better overall performance. Additionally, the simulation studies and real-time experimentation showed that the proposed controller is effective in controlling a DC servo system. Ahmed *et al.* [10] designed a fuzzy PID controller that uses self-tuning parameters and a neuro-fuzzy structure. The advantage of this approach is that the equation for the classical PID controller can be derived from rules composed of fuzzies. This makes the fuzzy PID controller similar to conventional controllers, which has advantages in terms of simulation performance. They evaluated the simulations performed with the proposed fuzzy PID controller in real-time applications. Implementations were carried out using real-time workshop software on MATLAB platforms. Sarabakha *et al.* [11] explained how the creation of control signals is affected by the footprint of uncertainty (FOU) characteristics. An analysis was conducted for single input IT2 fuzzy PID (SI-IT2-FPID) controllers by presenting the effect of the FOU parameters on control surface generation in order to improve the interpretability of specially structured interval type-2 (IT2) fuzzy logic controllers, more specifically a SI-IT2-FPID controller.

Hasan and Awad [12] presented a fuzzy PID controller to drive the motor using a Wi-Fi network to resolve the variable delay issue caused by the network's insertion in the feedback loop of a wirelessly driven DC motor. Both the TrueTime network simulator and the particle swarm optimization (PSO) algorithm, which are used to model the Wi-Fi network, are crucial for fine-tuning the controller. According to the findings, the suggested system can handle 7500 nodes on a network with a bandwidth share of 0.4 without experiencing any performance degradation. Ardeshiri *et al.* [13] used fractional order fuzzy PID controller optimized with a multiobjective PSO (MOPSO) algorithm for a robot manipulator with 2 links. Khan *et al.* [14] developed a self-tuning fuzzy PID controller (STFPID) to improve performance regarding an inverted pendulum in crane mode operation. Mehndiratta *et al.* [15] used a real-time 3 degree-off-reedom (DOF) helicopter testbed to evaluate their performance and presented experimental validation results demonstrating the design simplicity of a single input interval type-2 (IT2) fuzzy PID (FPID) controller.

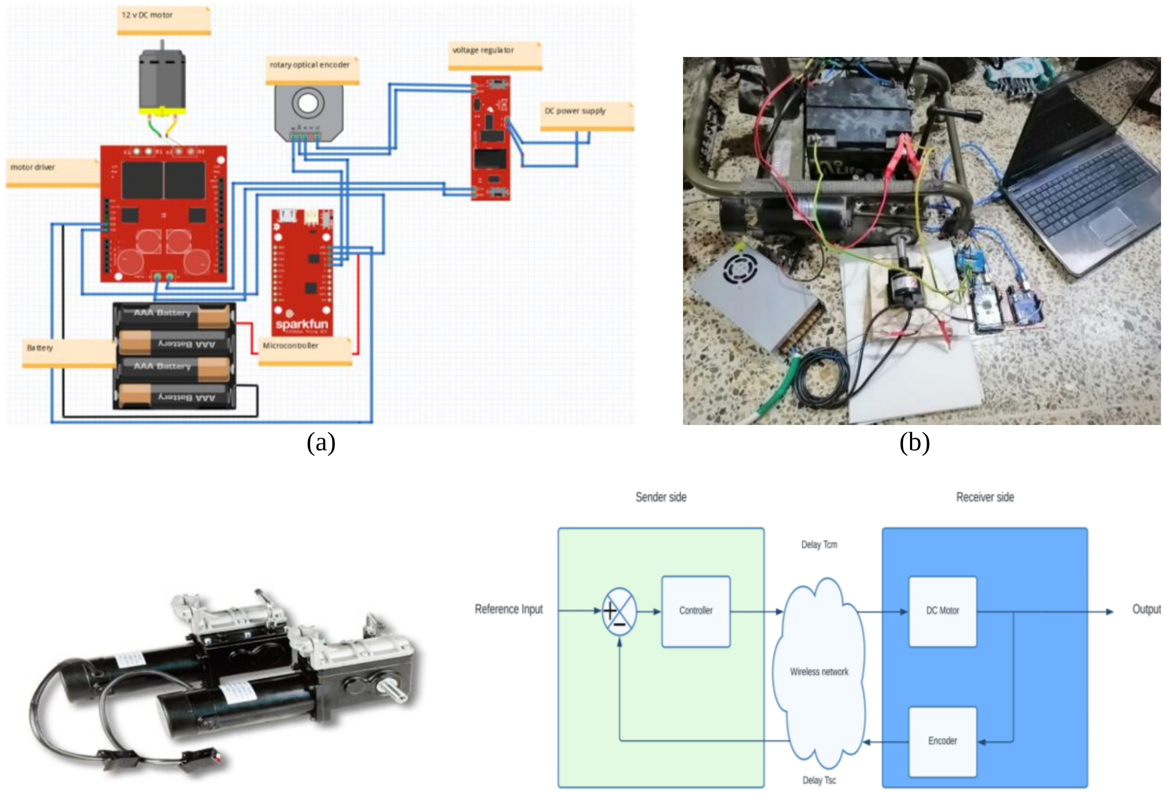
Singh *et al.* [16] designed a controller to stabilize the strong nonlinear coupling between the rotors of a helicopter system two DOF DC motor using an intelligent technique and achieve stability during the operation of the system. Kumbasar and Hagras [17] used an architecture with a cascaded control system, including the inner and outer control loops, to track the mobile robot's path. Shen *et al.* [18] presented an FPID controller to enhance the performance of a lower limb rehabilitation robot (LLRR) trajectory tracking. Sain and Mohan [19] modeled and designed a two-dimensional input FPID controller using space and center of gravity defuzzification. The newly developed FPID controller incremental control effort produced is found by adding the individual control efforts of the incremental FPI and incremental FPD controllers. Kumbasar and Hagras [20] used the big bang–big crunch optimization (BB–BC) technique to optimize the antecedent membership parameters of a controller which is an interval type-2 fuzzy PID using a cascaded control structure. Sahoo *et al.* [21] analytically studied the effect of using a type-2 fuzzy fractional-order PD-PI (T2FFOPDPI) controller in a microgrid system (MGS) for frequency regulation. Zhou *et al.* [22] to resolve the conflict between real-time online simulation and accuracy under various operating conditions, a real-time accurate equivalent circuit model (RAECM) of a pumped storage unit (PSU) was proposed via error compensation. An adaptively predicted fuzzy PID controller (APFPID) based on the RAECM was then used to get around the instability of a standard controller under no-load conditions with a low water head.

This paper focuses on implementing a real-time optimized fuzzy PID-controlled network-based remote DC motor. TrueTime software will be used to simulate the network while using node MCU, a low-cost internet of things (IoT) platform with open-source programming [23]. The microcontroller included firmware compatible with ESP 8266 Wi-Fi system on a chip (SoC) from esperssif systems and hardware that was based on the sep-12 module [23], [24] to move data between applications using the transmission control protocol/internet protocol (TCP/IP) protocol and logical ports, bypassing the conflicts that would normally occur when using different instrument performance simulator (IPS) for simulator and real-time systems.

2. RESEARCH METHOD

2.1. Real-time optimized fuzzy controlled wireless networked system

This paper proposes a real-time system consisting of a dg-158 type high-torque DC motor controlled by a fuzzy PID controller optimized using the PSO technique. Figure 1 illustrates the close-loop control system used; Figure 1(a) show fritzing schematic of the complete system setup, Figure 1(b) show high torque DC motor connected with the driver and optical rotary encoder, Figure 1(c) show the DC motor used, Figure 1(d) block diagram of the system with the network.



(a) (b)

(c) (d)

Figure 1. Different levels of the system (a) fritzing schematic of the complete system setup Figure 1, (b) high torque DC motor connected with the driver and optical rotary encoder, (c) dg-158 type DC motors, and (d) block diagram of the system with a network

2.2. Method

This paper proposes a methodology for the successful implementation of a real-time system, as illustrated in Figure 2. The methodology consists of hardware design, fuzzy PID controller implementation, TrueTime simulation, experimental data analysis, and result presentation. This approach provides a comprehensive framework for designing, implementing, and evaluating real-time systems for WNCS:

– Acquire the motor model by using PSO [12].
– Simulate the whole system using the real network simulator (TrueTime).
– Tune the FPID controller by using the PSO according to the calculated delay.
– Run the real-time system with the tuned controller using the TrueTime simulated network.

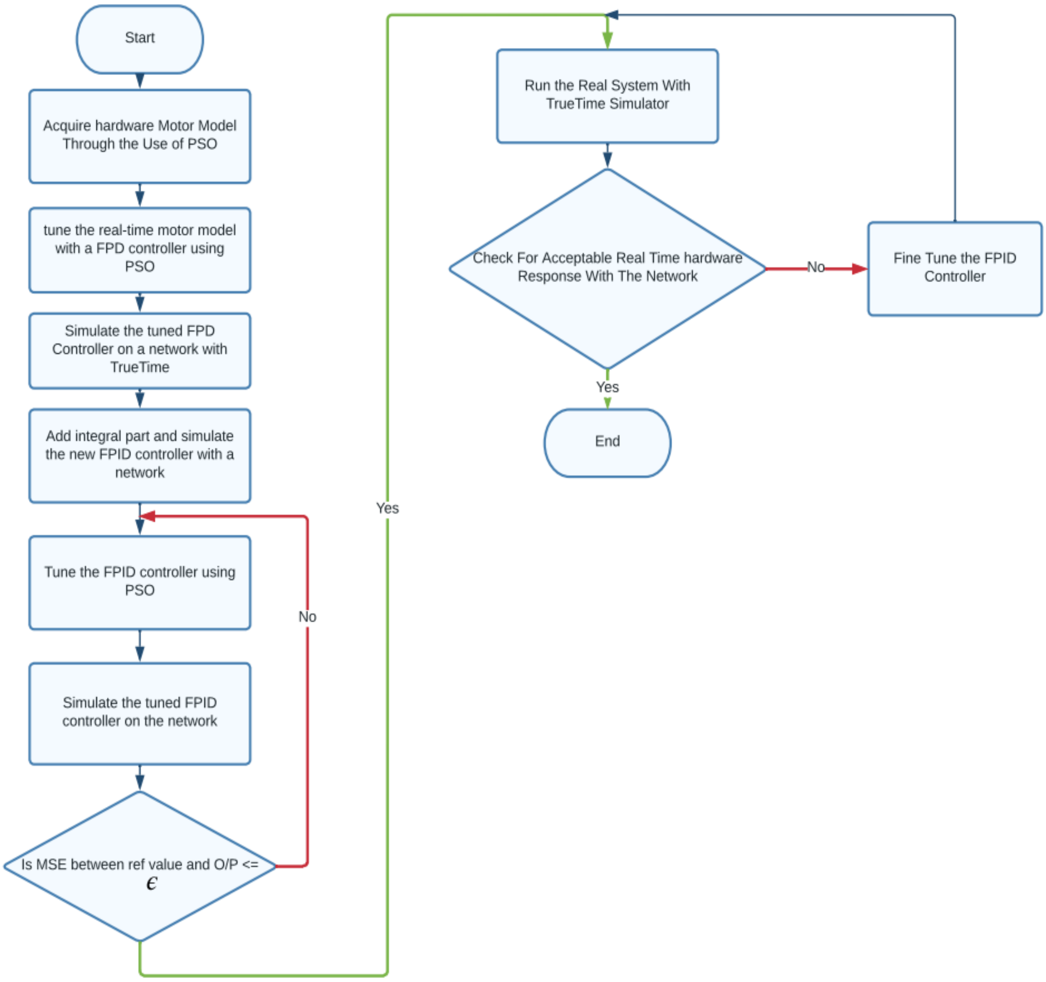


Figure 2. Flowchart of the methodology

The first step in this work is to evaluate the plant model (which is a DC motor with no data sheet), which we can achieve by using the PSO optimization technique. The objective function used to evaluate the PSO candidates is based on the mean squared error (MSE) criteria given by (1) and (2):

$$F = T_s \sum_{i=1}^{N} \sqrt{\theta_a^2(i) - \theta_m^2(i)} \tag{1}$$

$$\mathrm{N} = \frac{T_f}{T_s} \tag{2}$$

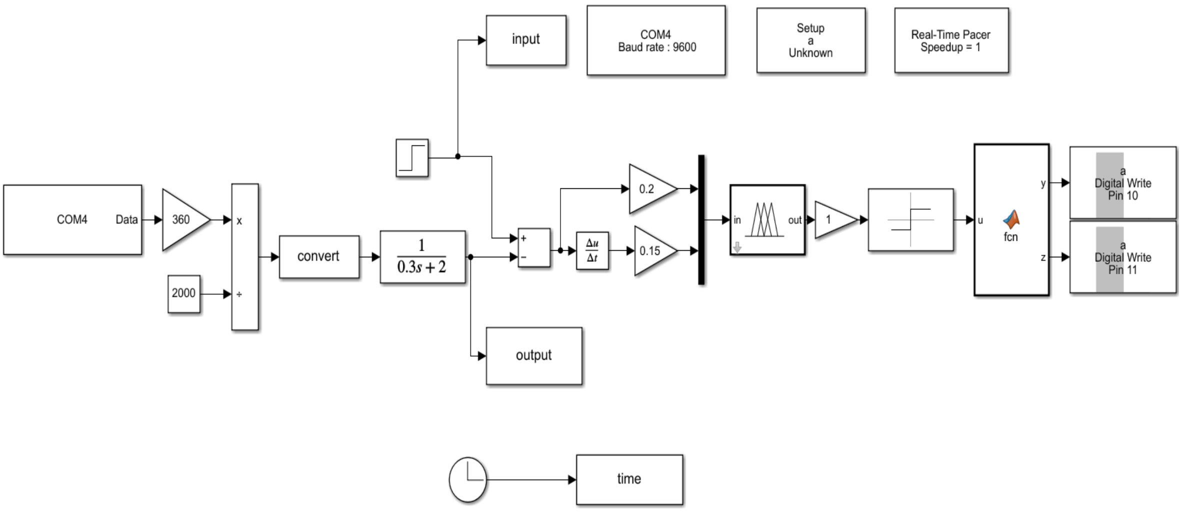
where T_s, T_f are the sampling time and final time, N is the number of samples and θ_a, θ_m are the angular positions for the modeled and the actual systems, respectively. It is well known that the DC motor can be modeled by a second-order system [25] hence (3):

$$\frac{\theta_m(s)}{\theta_r(s)} = \frac{K}{as^2+bs+c} \tag{3}$$

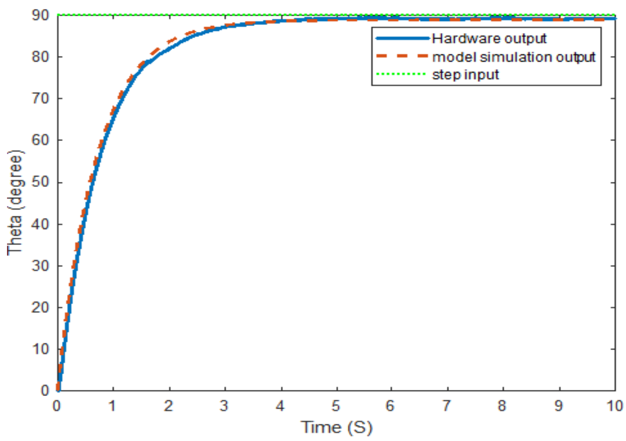
where k, a, b and c are the parameters that characterize the DC motor under investigation. The transfer function is found to be (4).

$$\frac{\theta_a(s)}{\theta_m(s)} = \frac{208}{s^2+180s+3.5} \tag{4}$$

This is acquired by using a closed loop system as shown in Figure 3, where Figure 3(a) is the MATLAB program used. The 2000 ppr (pulse per revolution) rotary optical encoder connected to the DC motor shaft provided the angle. A system with a fuzzy proportional derivative controller (FPD) is simulated to obtain the response and model of the DC motor as shown inf Figure 3(b).



(a)



(b)

Figure 3. Software part of the system (a) closed loop the controller uesd to communicate between the motor and encoder and (b) real-time DC motor angular position response of the system with a step input of 90 degrees

2.3. Network simulation and real-time system interface

Using the block diagram shown in Figure 4, the controller is tested on a network with actual hardware implementation. To do so there is a problem that must be solved first. The TrueTime simulation program does not work with direct serial communication with Arduino microcontroller IDE software to feed the encoder position to MATLAB directly.

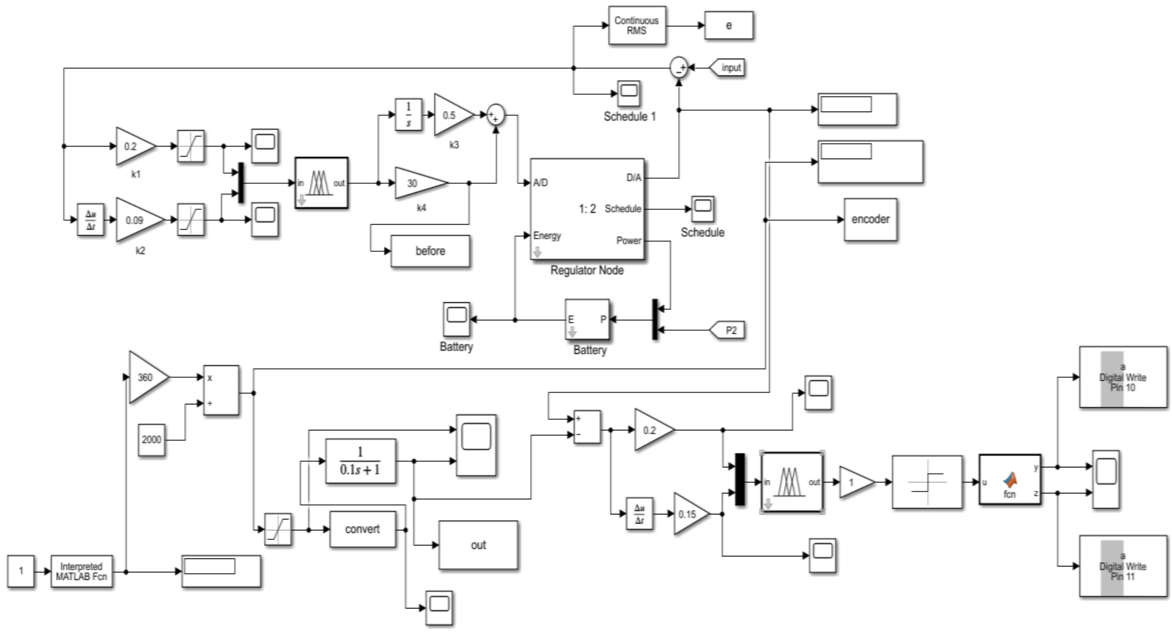


Figure 4. The subblock of the controller node in MATLAB along with microcontroller-MATLAB interface with DC motor driver

To solve this problem, the TCP/IP or user datagram protocol (UDP) protocols will be used with NODE MCU to link the two platforms. Serial data cannot be used in this case therefore we will connect ESP8266 which is a type of NODE MCU to COM3 or any available serial port with a 9600 baud rate to receive the encoder angle and then transfer it to a local server for reading from the MATLAB software client side using the IP address and port number which will be assigned to the MATLAB side. The host will be set to “localhost” and port number 1500 or any port within the 0 to 65535 computer logical port range. We need this procedure to bypass the serial port problem with TrueTime. Note that this method is very slow compared to the serial direct approach since it needs three programs and is not an interrupt routine.

3. RESULTS AND DISCUSSION

This paper optimized the fuzzy PID controller of the proposed system using the PSO technique with parameters shown in Table 1. The PSO algorithm was selected for its ability to handle complex optimization problems. The parameters were carefully chosen to ensure convergence to the optimal solution. The optimized controller can handle delays and uncertainties in WNCS.

Table 1. PSO parameters for controller coefficient calculation

Parameters	Number of birds	Birds step	The dimension of the problem	C_1	C_2	weight
Value	60	60	4	3.5	3.5	0.2638

Then we tested different sampling times and network load conditions to determine the response of the system. The system will be tested with a network of variable node numbers and sampling time while taking caution regarding bandwidth sharing. This might make the system unstable with a highly loaded network with a number of nodes higher than 240. More details about the system network conditions are discussed in the coming tables. Sampling times of 0.01 and 0.001 seconds will be tested. The packet length used will be calculated based on (2) as shown in (5):

$$P_l = (\mathrm{N} \times BW_{share} \times P_{max} \times \omega)byte \tag{5}$$

where P_l the total packet length in bytes, N is the number of nodes, BW_{share} use a fraction of the bandwidth that is currently occupied by other nodes on the network. If this number is high, then it will cause the network to be overloaded and vice versa, P_{max} fixed value for the maximum size of packet length that is 2304 bytes when using a Wi-Fi network, and w is a random number between 0 and 1 that determines the size of packets sent through the network.

3.1. Sampling time of 0.01 Sec with TCP/IP protocol

We will test our system with different parameters and using the TCP/IP and UDP protocols for data transfer from the encoder and to the motors. The simulation time is set to 10 sec, Table 2 shows the measured results of the simulated WNCS with different measurements regarding system characteristics (overshoot, settling time, rise time, and RTT), The steady-state response is achieved with different sampling times, as shown in Tables 2 and 3.

Table 2. Performance metrics for a network with T_s=0.01 Sec, BW_{share}=0.4, and TS_i =0.035 Sec

$T_s = 0.01$(Sec) BW_{share}= 0.4, TS_i= 0.035(Sec)			
No. of nodes in network	Overshoot %	Settling time (Sec)	Rise time (Sec)
200	0.0011	0.6216	0.3827
300	0.0016	0.6027	0.3636
400	2.8055	0.5836	0.3268

Table 2 presents the results of the system's response to different network sizes, and it demonstrates highly effective characteristics with settling times of less than one second and minimal overshoot across all network sizes. The settling time varies with larger networks, with the best performance observed in the 300-node system as shown in Figure 5, which had a settling time of 0.6027 seconds. This result is very similar to the 200-node system, with negligible differences in overshoot and settling time, as shown in Figure 5(a). These results indicate that the proposed controller design can effectively stabilize large-scale networks within a short settling time. It's worth noting that the designed controller was able to stabilize all the tested networks in under 10 seconds, demonstrating its effectiveness for controlling different network sizes. In the next step, we will test the system with a larger network and a bandwidth share of 0.7. Table 3 presents the response characteristics of the system.

Table 3. Performance metrics for a network with T_s=0.01 Sec, BW_{share}=0.7, and TS_i =0.035 Sec

$T_s = 0.01$(Sec) BW_{share}= 0.7, TS_i= 0.035(Sec)			
No. of nodes in network	Overshoot %	Settling time (Sec)	Rise time (Sec)
500	661.4231	4.1628	0.2283
600	591.8265	4.3847	0.2281
700	818.9193	5.1032	0.2281

Table 3 provides detailed performance metrics for the system's response to different network sizes. As shown in the table, for the 500-node network, the system experienced an overshoot percentage of 661.4231%, a settling time of 4.1628 seconds, and a rise time of 0.2283 seconds. The 600-node network had an overshoot percentage of 591.8265%, a settling time of 4.3847 seconds, and a rise time of 0.2281 seconds. Finally, for the 700-node network, the system had an overshoot percentage of 818.9193%, a settling time of 5.1032 seconds, and a rise time of 0.2281 seconds.

These results indicate that while the proposed controller design is effective in stabilizing the system for all tested network sizes, larger networks require longer settling times and exhibit higher overshoot percentages. However, even for the largest tested network with 700 nodes, the system was able to achieve stability in under 6 seconds. Overall, the results suggest that the proposed controller design is a promising solution for stabilizing large-scale networks in various applications. The test results for 0.7 bandwidth share are shown in Figure 5(b).

The reason for the 12-volt control signal in Figures 5(c) and 5(d) is that the motor operates on a 12-volt battery with a current of 17 amps. This voltage and current specification is a crucial factor that affects the motor's performance, such as its speed and torque. Therefore, the control system's design must consider the characteristics of the power source to optimize the motor's operation.

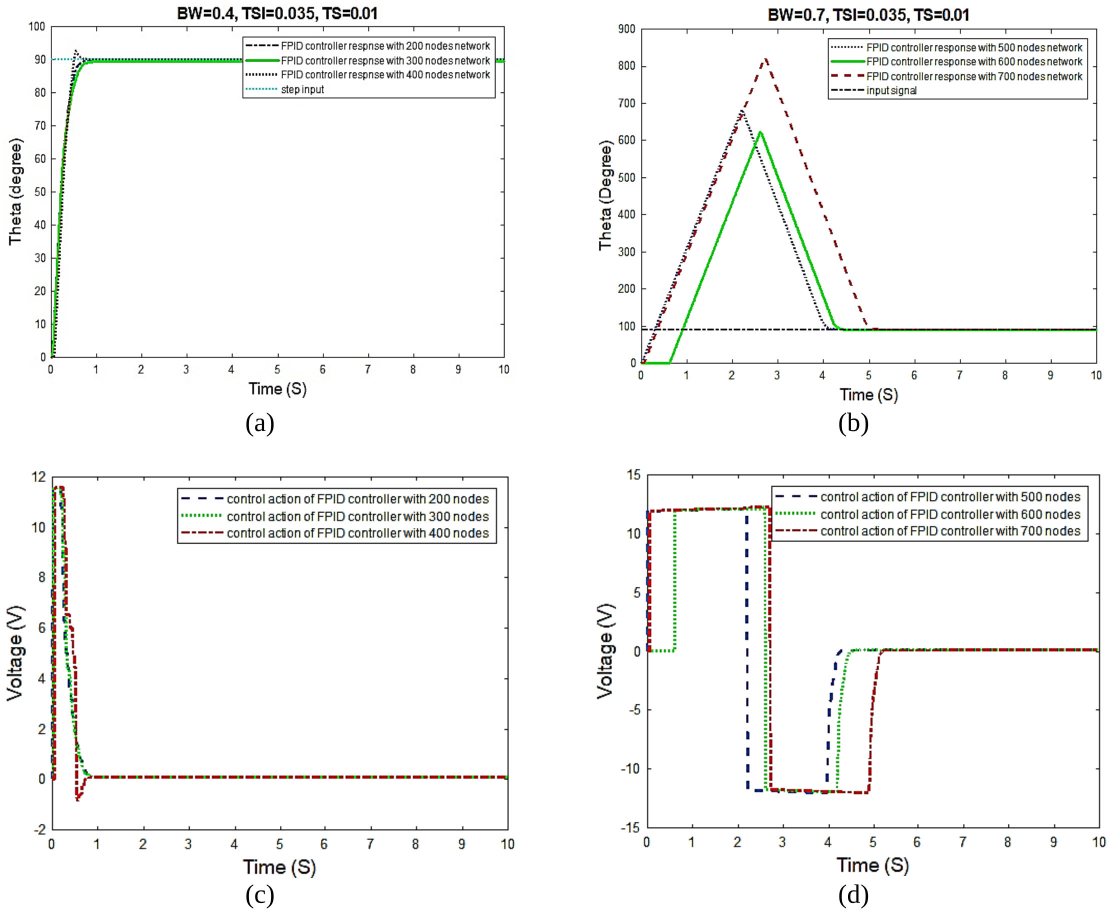


Figure 5. Motor response with different nodes variation using 0.01 sampling (a) DC-motor angular displacement using fuzzy PID controller with bandwidth share of 40%, (b) DC-motor angular displacement using fuzzy PID controller with bandwidth share of 70%, (c) 40% FPID controller control action, and (d) 70% FPID controller control action

3.2. Sampling time of 0.001 Sec with UDP protocol

To confirm the harmonic property of our design and assess compatibility between our system and the TrueTime program, which uses TCP, we will perform tests using UDP protocols with a sampling time of 0.001 seconds. These tests will be conducted with different numbers of nodes, as detailed in Table 4.

Table 4. Performance metrics for a network with T_s=0.001 Sec, BW_{share}=0.4, and TS_i =0.035 Sec

T_s = 0.001(Sec) BW_{share}=0.4, TS_i = 0.035(Sec)			
No. of nodes in network	Overshoot %	Settling time (Sec)	Rise time (Sec)
200	36.9490	5.4706	0.2670
300	37.3045	6.0320	0.3399
400	228.1813	7.5473	0.2655

The results of Table 4 show that for the 200-node network, the system experienced an overshoot percentage of 36.9490%, a settling time of 5.4706 seconds, and a rise time of 0.2670 seconds. The 300-node network had an overshoot percentage of 37.3045%, a settling time of 6.0320 seconds, and a rise time of 0.3399 seconds. The 400-node network had an overshoot percentage of 228.1813%, a settling time of 7.5473 seconds, and a rise time of 0.2655 seconds. These results suggest that the proposed controller design is effective in stabilizing the system for networks of different sizes, with overshoot percentages and settling times varying based on network size. Interestingly, for the 200-node and 300-node networks, the overshoot percentages were relatively low, while for the 400-node network, the overshoot percentage was considerably higher. Similarly, the settling time increased with larger network sizes, with the 400-node network experiencing the longest settling time of 7.5473 seconds. Overall, the results suggest that the proposed controller design is effective in stabilizing the system across a range of network sizes, with the overshoot and settling time varying based on the network size. However, even for the largest tested network with 400 nodes, the system was able to achieve stability in under 8 seconds. Table results are plotted in Figure 6 with Figure 6(a) showing angular displacement.

When comparing Table 5 to the previous tables, it is clear that the performance of the system for larger networks is much worse than for smaller networks. The overshoot percentage is much higher, and the settling time is significantly longer for the 500, 600, and 700-node networks compared to the smaller networks, indicating that the system becomes increasingly unstable as the network size increases. Additionally, the rise time for the 700-node network is extremely low, which indicates that the system is not able to respond to changes in the network quickly. Overall, the data in Table 5 demonstrates that while the system remains effective for smaller networks, its performance deteriorates significantly for larger networks, making it unstable and less effective. Table results are plotted in Figures 6(b)-6(d) show the control signal for the controller with 0.001 sec sampling time.

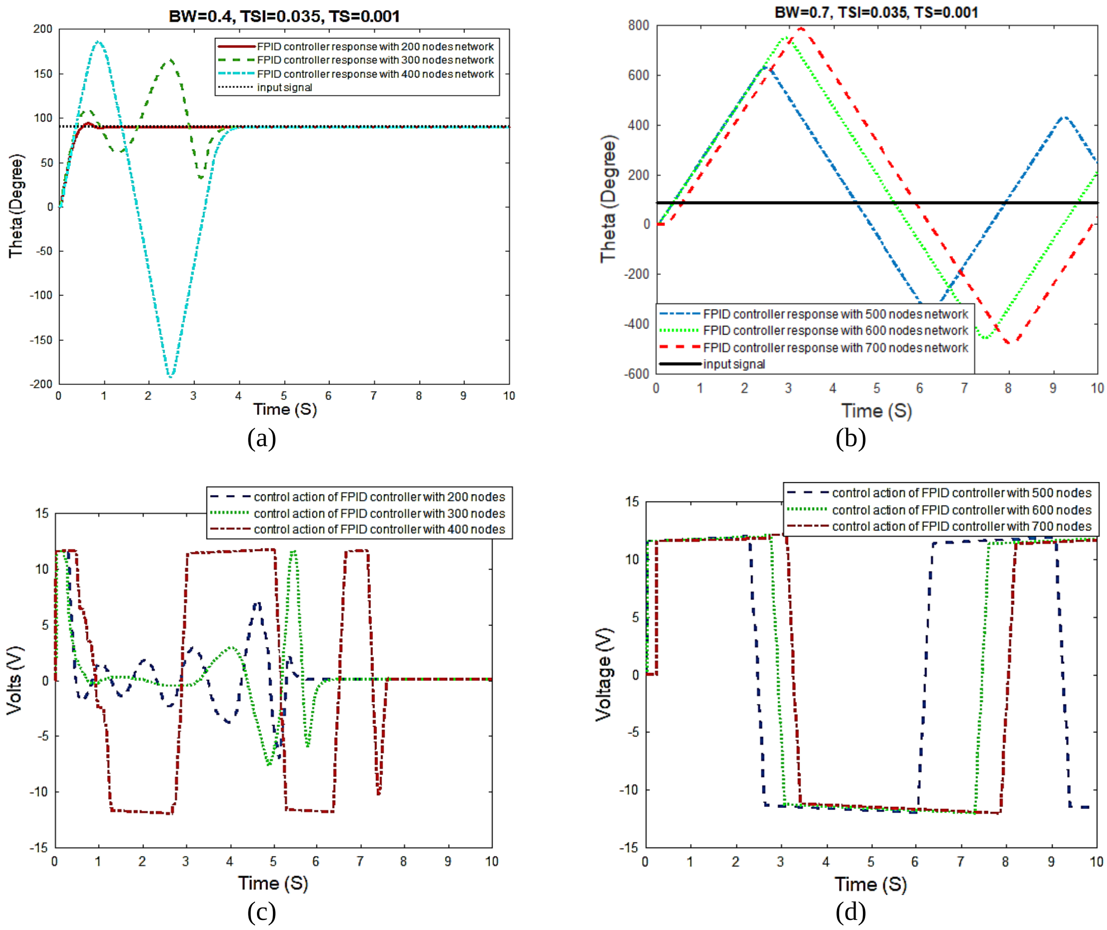


Figure 6. Motor response with different nodes variation using 0.01 sampling (a) DC motor angular displacement using fuzzy PID controller with bandwidth share of 40% and 0.001 T_s, (b) DC-motor angular displacement using fuzzy PID controller with bandwidth share of 70% and 0.001 T_s, (c) 40% FPID controller control action, and (d) 70% FPID controller control action

Table 5. Performance metrics for a network with T_s=0.001 Sec, BW_{share}=0.7, and TS_i =0.035 Sec

T_s = 0.001(Sec) BW_{share}=0.7, TS_i = 0.035(Sec)			
No. of nodes in network	Overshoot %	Settling time (Sec)	Rise time (Sec)
500	154.4384	9.9820	0.7361
600	256.3032	9.9845	0.6253
700	2.4578e+03	9.9977	0.0914

Results in Table 4 for sampling time of 0.001 sec and a bandwidth share of 0.4 As the number of nodes increased, the overshoot percentage increased significantly, with the settling time increasing from 5.4706 sec to 7.5473 sec. Table 5 taken for same sampling time and a higher bandwidth share of 0.7. As the number of nodes increased, the overshoot percentage increased significantly, with the settling time increasing from 9.9820 sec to 9.9977 sec, but this result is not accurate because the simulation is for 10 sec total time meaning that the system is unstable for any network higher than 400 nodes, this highlights the importance of carefully choosing the network parameters and controller settings in order to achieve stable and reliable performance in wireless networked control systems. Overall, these results demonstrate the system's real-time response on wireless networks with varying configurations. Table 6 show comparison between the different selected networks performance.

Table 6. Performance comparison for different network sizes

No. of nodes in network	Overshoot %	Settling time %	Rise time %
200	3295363.6364	779.5062	-30.2069
300	1896531.2500	898.3631	-6.4328
400	8029.5670	1290.2588	-18.9472
500	2037.1255	6989.7695	-68.0128
600	-57.2545	12943.6238	-64.0529
700	201.7522	20045.4215	-62.1285

The table demonstrate that the overshoot percentage difference shows a significant increase as the number of nodes increases, indicating that the larger networks tend to have more overshoot. The settling time percentage difference shows a mixed trend, with some values increasing and some decreasing as the number of nodes increases. The rise time percentage difference generally decreases as the number of nodes increases, indicating that larger networks tend to have faster rise times. The table also shows a negative value for overshoot difference and a positive value for settling time and rise time difference for the case where the number of nodes increased from 200 to 300, which suggests that increasing the number of nodes improved the network performance for these criteria. The table highlights the trade-offs involved in designing large-scale networks, where increasing the number of nodes can improve certain aspects of performance but may also lead to other performance degradation due to delays and other factors.

3.3. Wireless networked controlled hardware response

The FPID controller will be tested with the DG-158 type high-torque DC motor system using a 200 nodes network, a bandwidth share of 0.4, and a sampling time of 0.01 seconds as shown in Figure 7. These parameters were selected based on previous simulation results. The results of this test will provide valuable insights into the system's real-world performance and help identify potential areas for improvement. Figure 7 demonstrates that when a sampling time of 0.01 seconds and a bandwidth share of 0.4 were used in a 200-node network, the system's response closely matched the simulation results after filtering out noise and applying conditional commands with a microcontroller.

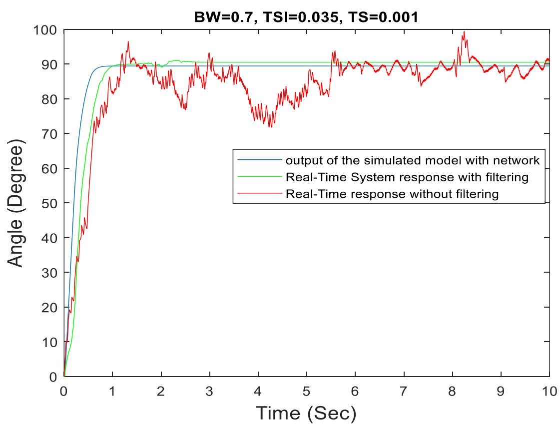


Figure 7. Real-time DC-motor angular displacement without and with filtering compared to the simulated model controlled through a wi-fi network

4. CONCLUSION

In this paper, a real-time optimized fuzzy PID-controlled wireless networked system with a cooperative TCP/IP and UDP protocols network simulator is designed and implemented. The delay in WNCS is the main problem that faces such systems and was solved here by using a powerful controller that can compensate for the occurring delay. The system which is a high torque DC motor was efficiently controlled with a powerful FPID controller tuned to operate on a network with PSO technique and was tested on a network by using the TrueTime simulator. To make this possible and unique from other research papers the system was tested in real-time while using the simulator which was possible by solving the IP conflict occurring in such cases by using NodeMCU to transfer the data to the simulator. The system was tested on a network with different values for nodes, sampling time, and bandwidth share. Based on the acquired results, we can draw several conclusions about the system being tested. Firstly, we can see that increasing the number of nodes in the network has a negative impact on the system's performance, resulting in higher overshoot, longer settling time, and longer rise time. When the number of nodes increases, the system's performance deteriorates. Secondly, we can observe that the choice of bandwidth share and sampling time has a significant impact on the system's performance. When the bandwidth share was 0.4, the settling time improves with the decrease in the sampling time, which is indicative of a better real-time response. However, when the bandwidth share was set to 0.7, we see that the settling time deteriorates as the sampling time decreases, which suggests that a longer sampling time may be necessary for stable operation in these conditions. Lastly, we can see that the system is unstable with shorter sampling time and high load in the system, as the overshoot is very high, and the settling time is very long. This indicates that further improvements to the system's design and control algorithms may be necessary to achieve stable operation. In conclusion, the tables present valuable insights into the performance of the system being tested, highlighting the impact of various factors such as the number of nodes, bandwidth share, and sampling time on the system's stability and real-time response.

REFERENCES

[1] P. Park, S. C. Ergen, C. Fischione, C. Lu, and K. H. Johansson, "Wireless network design for control systems: a survey," *IEEE Communications Surveys and Tutorials*, vol. 20, no. 2, pp. 978–1013, 2018, doi: 10.1109/COMST.2017.2780114.

[2] M. Andersson, D. Henriksson, and A. Cervin, "TrueTime 1.3— reference manual," *TrueTime 1.3-reference manual. Department of Automatic Control, Lund Institute of Technology*, 2016.

[3] W. Lee and M. Yoo, "Implementation of industrial wireless network based on IEEE 802.15.4e for real-time control system," *The Journal of Korea Information and Communications Society*, vol. 39B, no. 5, pp. 291–295, May 2014, doi: 10.7840/kics.2014.39b.5.291.

[4] C. Lu *et al.*, "Real-time wireless sensor-actuator networks for industrial cyber-physical systems," *Proceedings of the IEEE*, vol. 104, no. 5, pp. 1013–1024, May 2016, doi: 10.1109/JPROC.2015.2497161.

[5] M. Razfar *et al.*, "Wireless network design and analysis for real time control of launch vehicles," *In IEEE International Conference on Wireless for Space and Extreme Environments,* Nov. 2013, doi: 10.1109/WiSEE.2013.6737574.

[6] L. D. Liao *et al.*, "Design and validation of a multifunctional android-based smart home control and monitoring system," *IEEE Access*, vol. 7, pp. 163313–163322, 2019, doi: 10.1109/ACCESS.2019.2950684.

[7] S. Liu, B. Xing, and W. Zhu, "A fusion fuzzy PID controller with real-time implementation on a ship course control system," in *2015 23rd Mediterranean Conference on Control and Automation, MED 2015 - Conference Proceedings*, Jun. 2015, pp. 916–920, doi: 10.1109/MED.2015.7158875.

[8] R. R. D. Maity, R. K. Mudi, and C. Dey, "Real-time evaluation of an interval Type-2 fuzzy PID controller on servo position control system," *In IEEE International Conference on Wireless for Space and Extreme Environments,* Jan. 2018, doi: 10.1109/EAIT.2018.8470410.

[9] C. Dey, R. K. Mudi, and P. Mitra, "A self-tuning fuzzy PID controller with real-time implementation on a position control system," in *Proceedings - 2012 3rd International Conference on Emerging Applications of Information Technology, EAIT 2012*, Nov. 2012, pp. 32–35, doi: 10.1109/EAIT.2012.6407855.

[10] S. A. Ahmed, A. M. Taneva, M. G. Petrov, and I. K. Ganchev, "Self-tuning fuzzy pid: real time control applications," *In International Conference Automatics and Informatics*, 2013, pp. 1–79.

[11] A. Sarabakha, C. Fu, E. Kayacan, and T. Kumbasar, "Type-2 fuzzy logic controllers made even simpler: from design to deployment for UAVs," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 6, pp. 5069–5077, Jun. 2018, doi: 10.1109/TIE.2017.2767546.

[12] I. A. Hasan and O. A. Awad, "An optimized fuzzy logic controller for wireless network control system using PSO," *Iraqi Journal of Information and Communication Technology*, vol. 5, no. 1, pp. 1–15, Apr. 2022, doi: 10.31987/ijict.5.1.180.

[13] R. R. Ardeshiri, M. H. Khooban, A. Noshadi, N. Vafamand, and M. Rakhshan, "Robotic manipulator control based on an optimal fractional-order fuzzy PID approach: SiL real-time simulation," *Soft Computing*, vol. 24, no. 5, pp. 3849–3860, Jun. 2020, doi: 10.1007/s00500-019-04152-7.

[14] A. Khan, B. Chatterjee, and C. Dey, "Real time performance evaluation of a self-tuning fuzzy PID controller on an inverted pendulum in crane mode operation," in *2016 International Conference on Intelligent Control, Power and Instrumentation, ICICPI 2016*, Oct. 2017, pp. 22–26, doi: 10.1109/ICICPI.2016.7859666.

[15] M. Mehndiratta, E. Kayacan, and T. Kumbasar, "Design and experimental validation of single input type-2 fuzzy pid controllers as applied to 3 DOF helicopter testbed," in *2016 IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 2016*, Jul. 2016, pp. 1584–1591, doi: 10.1109/FUZZ-IEEE.2016.7737879.

[16] R. Singh, B. Bhushan, and A. Varshney, "Real-time simulation and control of helicopter systems," in *2019 International Conference on Power Electronics, Control and Automation, ICPECA 2019 - Proceedings*, Nov. 2019, vol. 2019-Novem, doi: 10.1109/ICPECA47973.2019.8975393.

[17] T. Kumbasar and H. Hagras, "A type-2 fuzzy cascade control architecture for mobile robots," in *Proceedings - 2013 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2013*, Oct. 2013, pp. 3226–3231, doi: 10.1109/SMC.2013.550.

[18] Z. Shen, J. Zhou, J. Gao, and R. Song, "Fuzzy logic based PID control of a 3 DOF lower limb rehabilitation robot," in *8th Annual IEEE International Conference on Cyber Technology in Automation, Control and Intelligent Systems, CYBER 2018*, Jul. 2019, pp. 818–821, doi: 10.1109/CYBER.2018.8688089.

[19] D. Sain and B. M. Mohan, "Modeling, simulation and experimental realization of a new nonlinear fuzzy PID controller using center of gravity defuzzification," *ISA Transactions*, vol. 110, pp. 319–327, Apr. 2021, doi: 10.1016/j.isatra.2020.10.048.

[20] T. Kumbasar and H. Hagras, "Big bang-big crunch optimization based interval type-2 fuzzy PID cascade controller design strategy," *Information Sciences*, vol. 282, pp. 277–295, Oct. 2014, doi: 10.1016/j.ins.2014.06.005.

[21] G. Sahoo, R. K. Sahu, N. R. Samal, and S. Panda, "Analysis of type-2 fuzzy fractional-order PD-PI controller for frequency stabilisation of the micro-grid system with real-time simulation," *International Journal of Sustainable Energy*, vol. 41, no. 5, pp. 412–433, May 2022, doi: 10.1080/14786451.2021.1928131.

[22] J. Zhou, Z. Zhao, C. Zhang, C. Li, and Y. Xu, "A real-time accurate model and its predictive fuzzy PID controller for pumped storage unit via error compensation," *Energies*, vol. 11, no. 1, p. 35, Dec. 2018, doi: 10.3390/en11010035.

[23] A. Shrivastava, S. J. S. Prasad, A. R. Yeruva, P. Mani, P. Nagpal, and A. Chaturvedi, "IoT based RFID attendance monitoring system of students using Arduino ESP8266 & Adafruit.io on defined area," *Cybern Syst*, vol. 0, no. 0, pp. 1–12, 2023, doi: 10.1080/01969722.2023.2166243.

[24] M. Z. H. Zim, "TinyML: Analysis of Xtensa LX6 microprocessor for Neural Network Applications by ESP32 SoC," *arXiv:2106.10652*, 2021.

[25] K. Ogata, *Modern control engineering*. Prentice-Hall, 2010.

BIOGRAPHIES OF AUTHORS

Yousif Safaa Alobaidy was born in Baghdad, Iraq, in 1997. He received his B.Sc. in System Engineering, the College of Information Engineering at Al-Nahrian University, Iraq in 2018. Since 2019 he worked in different engineering fields including Medical Devices installation and maintenance and 4th generation mobile services BSS installation and repair in Iraq. He is currently a candidate for an M.Sc. at the Department of System Engineering, College of Information Engineering from Al-Nahrian University, Iraq, and his research interests include robotics, I2C communication, control, systems, and wireless network control. He can be contacted at email: yousif_saffa97@yahoo.com.

Osama Ali Awad is currently an associate professor at the information engineering college, Al-Nahrain University. He received his B.Sc. degree in control and systems engineering, from the University of Technology (UoT), Iraq in 1978, his M.Sc. degree in control engineering from UoT, Iraq in 1982 and received his Ph.D. degree in control and automation from UoT, Iraq in 2015. His main areas of interests include control of nonlinear dynamical systems, MIMO systems, soft computing, wireless sensor networks, biomedical control systems and networked control systems. He can be contacted at email: usamawad@nahrainuniv.edu.iq.