Android Application Design with MIT App Inventor for Bluetooth Based Mobile Robot Control

Ahmet Top · Muammer Gökbulut

Accepted: 8 May 2022 / Published online: 1 June 2022
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2022

Abstract
In this study, a Bluetooth-based Android application interface is developed to perform a manual and automatic control of a four-wheel-driven mobile robot designed for education, research, health, military, and many other fields. The proposed application with MIT App Inventor consists of three components: the main screen, the manual control screen, and the automatic control screen. The main screen is where the actions of the control preference selection such as manual control and automatic control and the Bluetooth connection between the mobile robot and Android phone occur. When the robot is operated manually for calibration or manual positioning purposes, the manual control screen is employed to adjust the desired robot movement and speed by hand. In the case of the need for automatic motion control, the desired robot position and speed data are inserted into the mobile robot processor through the automatic control screen. At the first stage of the work, the proposed Android application is developed with the design and block editors of the MIT App Inventor. The compiled application is then installed on the Android phone. Next, the communication between the Arduino microcontroller used for the robot control with the Bluetooth protocol and the Android application is established. The accuracy of the data dispatched to the Arduino is tested on the serial connection screen. It is validated that the data from the Android application is transferred to Arduino smoothly. At the end of this study, the manual and automatic controls of the proposed mobile robot are performed experimentally and success of the coordination between the Android application and the mobile robot are demonstrated.

Keywords Android application · MIT App Inventor · Bluetooth · Robot control

Ahmet Top
atop@firat.edu.tr

Muammer Gökbulut
mgokbulut@firat.edu.tr

1 Department of Electrical and Electronics Engineering, Faculty of Technology, Firat University, Elazığ, Turkey
1 Introduction

Along with the rapidly developing technology, robot technology shows great development. The workforce needed in jobs where human power is not sufficient or in environments that threaten human health is met by robot technologies [1]. While designing robots, it is important not only to provide functionality but also to provide ease of use. For this reason, it would be more appropriate to use a smartphone for robot control in terms of both cost reduction and ease of use. Nowadays, smartphones have almost become a part of the human body. Therefore with the developing technology, more powerful, faster and more capable smartphones are produced daily. In addition, they make human life much easier thanks to features such as Wi-Fi, bluetooth, GPS, camera and various sensors [2]. With the applications developed using these features, they are used for various purposes by communicating with objects as well as meeting the personal needs of people. These devices, produced by different companies, have different operating systems such as Android, IOS, Windows OS, Symbian OS. Android is the most used operating systems among them. Therefore, applications developed Android-based appeals to a wider audience.

MIT App Inventor, one of the development environments used to design Android applications, was first provided by the Google and is now maintained by the Massachusetts Institute of Technology (MIT) [3]. MIT App Inventor is a web-based and free-access development platform that provides a graphical interface to users, where fully functional Android applications can be made and designed [4, 5]. It uses a block-based programming language built on Google Blockly [6] and inspired by languages such as StarLogo TNG [7] and Scratch [8, 9]. MIT App Inventor consists of two parts. These are the designer and the block editor. The designer is the part where the screens and contents of the application are created. The block editor is an environment where the designer can visually edit the logic of their application with color-coded blocks that interlock like puzzle pieces to explain the program. In the last decade, besides applications such as automation systems [10–13], air pollution monitoring systems [14], mobile phone applications [15], education [16, 17], health [18], machine monitoring and control [2, 19] Android applications are also used for robotic systems [20–33].

Rajpar et al. designed a reconfigurable articulated robot with 3 degrees of freedom to perform chose and place tasks. While providing the movements of the robot consisting of 4 motors with Arduino, they sent the manual control commands from the Android platform they developed [20]. Nádvorník and Smutny performed the manual control of the mobile robot using bluetooth technology with an Android application that allows interactive robot control with image or sound. With the information they received from the sensors placed in front of the robot, they observed the distance of the robot to the obstacles through the application [21]. Aldhalemi et al. realized real-time control of the two-wheeled self-balancing robot by connecting with a smart phone [22]. Haripriya et al. carried out the control of a robot vehicle by transferring the voice commands sent with the Android application to the Arduino via bluetooth [23]. Erşahin and Sedef performed remote robot control with wireless technology. They installed the designed Android application on a tablet. They sent the images took from the camera placed on the robot to the tablet via Wi-Fi and sent the direction and speed values set from the tablet to the robot [24]. Ramya and Palaniappan performed a web-based remote control of a robot equipped with a motion sensor, camera and gas sensor for environmental safety. They completed successfully three tasks in the interface designed with Micro-soft Visual Basic. These are transferring camera images to the interface, detecting gas
leakage and live intrusion, and manually controlling the robot. [25]. Meteab et al. carried out the control of a robot vehicle containing Arduino microcontroller and bluetooth module with Android smartphone application [26]. Qadri et al. designed a robot with a 3D printer to measure carbon monoxide, temperature and humidity in mines where working conditions are dangerous and difficult. They developed an Android application for the control of this robot. The results of the measurements examined with a smart phone and they achieved 100% accuracy up to 10 m distance [27]. Papcun et al. communicated between mobile robot Khepera III and mobile device using bluetooth and Wi-Fi interface. They applied face recognition by sending the image taken from the camera on the robot to the mobile device [28]. Eren and Doğan realized the design and production of a high-performance and low-cost vacuum cleaner robot that can be controlled with a smartphone. In the software design of the robot, they designed an Android application for a bluetooth-based remote control. With the application, the operation of the vacuum and brush motor can be controlled while the robot moves [29]. Kırlı et al. who created the environment map with a mobile robot with six wheels and differential drive, made mobile robot operations and Android device developments using the robot operating system (ROS). While the camera image and the calculated map information taken from the mobile robot are transferred to the Android device, the data controlling the driving of the robot are transferred from the Android device to the robot [30]. Molnar et al. presented a study on the design and control of a mobile robot that maintains its balance by standing on two wheels. They made PID control using the ESP8266 microcontroller. By designing a mobile application with MIT App Inventor, they provided control over a mobile device and on a personal computer using the web interface [31]. Varga et al. have designed a mobile application for the Robot Soccer game that performs direction and motion control with an accelerometer by choosing between players 1 and 2 via bluetooth [32]. Fahmidur et al., with the Mobizen application, mirrored the screen of the Android phone to the computer and performed the robot control with bluetooth [33].

In this study, an Android application is developed to perform manual and automatic control of a 4-wheel drive mobile robot, whose design and prototype are realized. The interface and programming of the application are designed with the designer and block editor in MIT App Inventor. In studies related to this subject in the literature, mobile robots can only be controlled manually with the designed Android applications. This designed application offers both manual control and automatic control options. The application interface consists of three parts. The first part is the main screen seen when the application is opened, the second part is the manual control screen and the third part is the automatic control screen. On the main screen, it is possible to have a bluetooth connection between the robot and the smart phone/tablet, closing the application and send information for resetting the microcontroller card. It is also possible to pass to other screens. On the manual control screen, there are direction, speed adjustment, led and fan buttons to manually control the robot. On the automatic control screen, position reset and targeted axis and speed information are sent to the robot for automatic control of the robot. After the designed application was compiled, it was installed on the smart phone. Arduino Due microcontroller and HM-10 bluetooth module were used to test the accuracy of the information sent to the processor connected via bluetooth when the buttons in the application are pressed. By providing a bluetooth connection between the application loaded on the smartphone and the Arduino, the data sent from the application was observed on the Arduino Integrated Development Environment (IDE) serial screen and its accuracy was proven. In line with the data obtained, real-time manual control of the mobile robot with the Android application and automatic control in accordance with the references were carried out with the
program uploaded to the Arduino microcontroller, and the graphs of the robot control performances were given.

2 Mobile Robot

The mobile robot shown in Fig. 1 has a 4-wheel structure, each of which is connected to a 12 V DC motor, and moves with a differential driving system. Differential driving technique is a driving style based on having different speeds two of the four independent wheels on the common axis of the mobile robot. If the two wheels on the right and the two wheels on the left have different speeds, the vehicle turns to one side. If the wheel speeds have the same speed and direction, the vehicle go straight.

Robot power requirement is provided by the LiPo battery. A boost converter is used to stabilize the voltage value of the motors. The speed control of the motors is realized with two Sparkfun Monster Moto Shields as Pulse Width Modulation (PWM) controlled. These PWM values are sent by the Arduino Due microcontroller, where peripherals such as fan and headlights are also controlled. The data sent from the application installed on the smartphone is transferred to the Arduino with HM-10 BLE (Bluetooth Low Energy) bluetooth 4.0. According to these transferred values, motors, fans and headlights are controlled. The block diagram of the mobile robot is given in Fig. 2.

The connections between blocks in here is P: power, PD: PWM and direction E: encoder and C: Control.

3 Android Application Design

Since the mobile robot is desired to be controlled in two different ways, manual and automatic, the Android application interface is divided into three parts to both reduce complexity and provide visuality. These parts are Main screen, manual control screen and automatic control screen.

![Fig. 1 Mobile robot a front view; b rear view](image-url)
3.1 Main Screen

It is the screen that appears when the application is first run. On this screen, the bluetooth connection between the robot and the Android device is provided. In addition, there are selection buttons to pass to other parts, reset and application close button. Figure 3 shows the main screen interface and logic blocks.

The application communicates with the robot with the bluetooth protocol. Therefore, the one needs to know the device’s bluetooth address and connection status. In Fig. 4, the bluetooth connection workflow diagram and the blocks that perform this flow are given.

First of all, to find the device to be connected, scanning is done, selection is made from the list and then the connect button is pressed. If the connection is completed successfully, the ‘connect’ text turns green and the ‘connection established’ text appears at the bottom. If it fails, a ‘disconnected x’ warning appears in the status section. After the connection is established, the manual control button can be pressed to transition to the manual control screen, and the automatic control button can be pressed to transition to the automatic control screen. If these buttons are pressed without establishing a connection, the transition is provided, but data cannot be sent. Figure 5 shows the flowchart and logic blocks of these buttons.
Fig. 3  a Start screen logic blocks and b Screenshot
If the microcontroller software is to be restarted, the reset button can be pressed, if the application is to be closed, the exit button can be pressed. Thanks to the information to be sent to the device with the reset button, the software or hardware reset can be performed. When the exit button is pressed, a warning message is displayed to the user. If the user selects the ‘yes’ option, the application will be closed, if the user selects the ‘cancel’
option, the text will disappear and no action will be taken. The work flow chart and blocks of these buttons are shown in Fig. 6.

3.2 Manual Control Screen

It is the screen where the manual control of the mobile robot is provided. On the manual control screen given in Fig. 7, there are direction buttons, speed adjustment buttons and
indicators, return button, fan and headlight buttons. With these buttons, the robot can be moved in the desired direction, its speed can be adjusted and the fan and leds on it can be turned on and off.

In order to make the robot movement more stable and sensitive, the direction buttons are made with push-and-drop feature. When these buttons are pressed, the PWM value and direction data set in the application are sent to the microcontroller and the robot movement is provided. Since the direction information is not sent when it is released, the motors stop by their own inertia, according to the program written in Arduino. With these direction buttons, the
robot can move forward, backward, right forward, left forward, right, left, left back and right back. The PWM value sent to the robot with all direction buttons is the same, only the direction value changes. While the PWM value coming from the application is applied equally to all motors in the forward and reverse directions in the Arduino microcontroller, this PWM value is applied to the right and left motors at different rates in different directions. In this way, differential driving is provided. When the stop button shown in Fig. 7 is activated, the PWM
value and speed value on the screen are reset and the motors stop by braking in line with the information sent to the Arduino. Figure 8 shows the flowchart and logic blocks of the direction buttons and the stop button.

The speed of the robot is adjusted by changing the PWM value sent to the robot with the speed increase/decrease buttons. In the stage setting at the top of the screen, the increase/decrease rate of the PWM can be changed. If no change is made to the stage value, the PWM value increases by 1 by 1. While the PWM value is displayed at the top of increase but-
ton, the approximate speed value calculated according to the motor speed and wheel diameter can be seen among these buttons. The work flow chart and logic blocks of the speed increase–decrease buttons are shown in Fig. 9. Since the PWM signal is used as 16 bits in the microcontroller, it varies between 0 and 65,500 values. However, in order to limit the motor voltage to a certain value, the PWM upper limit is chosen as 51,500 and where the no load rotation speed at a maximum of 88 rpm. The motor speed calculated based on the PWM values adjusted from the increase/decrease buttons was considered equal to the robot speed and was calculated according to Eq. 1.

\[ V = \frac{88}{60} \times 2\pi r \times \text{pwm} \times \frac{1}{51500} \text{m/s}, \]  

(a) (b)

Fig. 8 a Flow chart and b logic blocks of direction buttons and stop button
where $V$ is the motor speed, $r$ is the wheel radius of the robot and $pwm$ is the entered pulse width modulation value.

In Fig. 10, flow charts and logic blocks of the return, fan and led buttons on the manual control screen are given. When the back button is pressed, the application returns to the main screen. There are four LEDs and one fan on the robot. While the LED button turns on and off the LEDs attached to the robot for lighting purposes, the fan button starts and stops the fan installed for cooling.

Since the LEDs and fan buttons work in the same way, they are shown together in the flow chart.

### 3.3 Automatic Control Screen

In the mobile robot processor, the speed and position of the robot can be controlled with algorithms such as PID, fuzzy, artificial neural networks, and soon. After the required algorithms are loaded into the robot processor, what needs to be done is to enter the target speed and position information for the robot. This process is performed on the automatic control screen of the Android application. As seen in Fig. 11, there are position
reset button, target position and speed entry, delete and confirm button and back button on this screen.

While performing automatic control, it may be necessary to reset the previous position information in the place where the robot is located and start from the origin point. For this reason, before entering the targeted axis values, information is sent to the Arduino with the ‘reset position’ button to reset the robot’s position. The x and y axes that the robot wants to go are entered in meters and the reference speed in m/s by the user. Then, when the OK button is pressed, these data are sent to the microcontroller and these values are processed as a reference. In case of entering the wrong value, the values can be re-entered by pressing the delete button. When the back button is pressed, it returns to the main screen. The flow chart and blocks of the automatic control screen are shown in Fig. 12.
Fig. 10  a Flow charts of the return, fan and led buttons; b Back button logic blocks, c Fan and leds logic blocks
Fig. 10  (continued)

Fig. 11  Automatic control screen
4 Experimental Results

When the application design was completed, the compilation process was done and an .apk file of 5.5 MB was created. For the functional test of the application, the program with the workflow given in Fig. 13 was written on the Arduino Due microcontroller and a bluetooth connection was made between the smartphone and Arduino as in Fig. 14.

After realising a bluetooth connection between the smartphone and the Arduino microcontroller, the manual control screen in the application interface was switched and various buttons were pressed on this screen. When the buttons in the Android application are
pressed, the values sent to the microcontroller were transferred to the serial port of the Arduino IDE program and examined. For screen image quality, screenshots were taken after pressing the buttons in the application. The green led and fan buttons indicate that the
on signal is sent, while the red ones indicate that the led/fan is sending the off signal. If the direction and speed buttons appear fainter than other buttons, it means that that button is pressed. As seen in Fig. 15, the value that should be sent via bluetooth and the value read
on the Arduino IDE serial screen are the same. This shows that the values sent from the application are successfully transferred to the microprocessor.

Figure 16 shows the speed values of each motor of the four wheel drive mobile robot in Fig. 1, which is moved in different directions in line with the information received from the application. In Arduino, all motors are programmed to go at equal speed while the robot is moving forward. As the robot moves in the right or left direction, the motor speeds are changing. 20,000 PWM values were sent to the robot from the application and the forward direction, right forward and left forward buttons were pressed respectively. While the motor moves forward in 0–4.5 s, it moves in the right forward direction in 6–10.5 s and in the left direction in 12.5–17 s. While switching between the buttons in the application, only the PWM value goes to the microcontroller. When there is no direction information in the program written in Arduino, the motors stop with their own inertia. Therefore, there was a
Fig. 15  Test results: a PWM = 0, forward direction selected; b PWM = 170, forward direction selected; c PWM = 0, led on; d PWM = 300, Stop button pressed; e PWM = 0, fan on; f Automatic control active x axis = 23 m, y axis = 63 m, ref. speed = 0.5 m/s
decrease in speeds at 4.5–10 s and 10.5–12 s. Since braking was done by pressing the stop button in 17 seconds, all four motors stopped suddenly.

In Fig. 17, the application screenshot of the movement in the 0–4.5 s interval is shown. The PWM value was determined to 20,000 and was sent to the microcontroller with forward button. The speed value calculated with this PWM value according to the formula in Eq. 1 is 0.37 m/sec. In Fig. 16, 34 rpm was measured for this time interval. When this value was converted to the unit of m/s with Eq. 2, it is seen that the robot speed in this range was 0.37 m/sec.

\[
V = V_{rpm} \times \frac{2\pi r}{60} \times \frac{1}{60} = 34 \times 2\pi 0.105 \times \frac{1}{60} = 0.37 \text{ m/sec},
\]

where \(V_{rpm}\) is the measured speed of motors and \(r\) is the wheel radius of robot. The PID control algorithm, whose block diagram is given in Fig. 18, is used to perform automatic speed and position control of the mobile robot. As seen in Fig. 19, these values were sent to the Arduino by entering the reference y-axis as 2 m, x-axis as 0 m and the speed as
Fig. 16 Changes of Motor speeds according to robot movement. The speed change graph of the robot moved in in Fig. 16. It is seen that the robot moves effectively in with the data taken from the smartphone in different directions.

Fig. 17 Manual control screenshot
0.22 m/s. According to Eq. 3, the robot must complete this task in 9.09 s. When the robot position and speed graphs given in Fig. 20 are examined, it is seen that the automatic movement is performed successfully.

\[
t = \frac{x}{V} = \frac{2}{0.22} = 9.09 \text{ s}
\]

In Eq. 3, \(x\) is the position, \(V\) is the speed and \(t\) is the time. In the robot automatic control diagram, \(K_p\), \(K_i\) and \(K_d\) are proportional, integral and derivative PID coefficients, respectively and \(e\) is the error signal.

As can be seen in Fig. 20, the robot position increased linearly and the robot 2-m reference position reached in 9 s with an error of 0.09 s. After 0.02 m/s overshoot at the start of the move, the robot was reached the steady state in 0.26 s and converge to the reference value.
Fig. 19 References sent from the automatic control screen

Fig. 20 Mobile robot: a position and b speed control
5 Conclusion

In this study, an Android application was developed with MIT App Inventor for bluetooth-based control of a 4-wheel-drive mobile robot. While the application interface was created in the MIT App Inventor designer, it was made with the drag and drop method in the block editor. When the application installed on the phone is run, the bluetooth connection is provided on the main screen that opens and the transition to other screens can be selected.

Arduino Due microcontroller was used to test that the values to be sent from the application are sent correctly when the buttons are pressed. A connection was established between the HM-10 Bluetooth module attached to the Arduino and the smartphone. In the program written to receive data in Arduino, the values sent with the application were transferred to the serial screen of the Arduino IDE program and examined and it was shown that the values were transferred as planned.

In line with the information sent to the Arduino, manual and automatic control of the four-wheeled mobile robot was carried out. The automatic speed and position control of the mobile robot was controlled with PID in line with the reference values sent from the application to the robot microcontroller. In the given graphics, it is presented that the mobile robot is controlled effectively manually and successfully captures the reference values with automatic control. The designed application can be installed on different Android-based phones or tablets, it can connect with any processor used in bluetooth protocol and can be used on different robots according to the program that the user will write on the microcontroller.

Funding This study was supported by Firat University Scientific Research Projects Unit with the project number TEKF.19.07.

Data Availability All data generated or analysed during this study are included in this published article.

Code Availability Can be provided if requested personally through corresponding author mail.

Declarations

Conflict of interest Authors declare no conflict of interest.

References

1. Akilan, T., Chaudhary, S., Kumari, P., & Pandey, U. (2020). Surveillance robot in hazardous place using IoT technology. 2020 2nd International conference on advances in computing, communication control and networking (ICACCCN) (pp. 775–780). IEEE. https://doi.org/10.1109/ICACCCN51052.2020.9362813
2. Mikolajczyk, T., Fuwen, H., Moldovan, L., Bustillo, A., Matuszewski, M., & Nowicki, K. (2018). Selection of machining parameters with Android application made using MIT App Inventor bookmarks. Procedia Manufacturing, 22, 172–179. https://doi.org/10.1016/j.promfg.2018.03.027
3. Hong, S., & Hwang, Y. (2020). Design and implementation for iort based remote control robot using block-based programming. Issues in Information Systems, 21(4), 317–330.
4. Pokress, S. C., & Veiga, J. J. D. (2013). MIT app inventor: Enabling personal mobile computing. arXiv preprint arXiv:1310.2830
5. Colter, A. J. (2016). Evaluating and improving the usability of MIT App Inventor (Doctoral dissertation, Massachusetts Institute of Technology).
6. Fraser, N. (2013). Blockly: A visual programming editor. Retrieved from https://code.google.com/p/blockly

7. Begel, A., & Klopfer, E. (2007). Starlogo TNG: An introduction to game development. *Journal of E-Learning*, 53(2007), 146.

8. Maloney, J., Resnick, M., Rusk, N., Silverman, B., & Eastmond, E. (2010). The scratch programing language and environment. *ACM Transactions on Computing Education (TOCE)*, 10(4), 1–15.

9. Amoran, A. E., Oluwole, A. S., Fagorola, E. O., & Diarah, R. S. (2021). Home automated system using Bluetooth and an android application. *Scientific African*, 11, e00711. https://doi.org/10.1016/j.sciaf.2021.e00711

10. Raheem, A. K. A. (2017). Bluetooth based smart home automation system using arduino UNO microcontroller. *Al-Mansour Journal*, 27, 139–156.

11. Liu, Y., & Uttra, D. R. A. (2020). Bluetooth based smart home control and air monitoring system. *International Journal of Advanced Research in Engineering and Technology (IJARET)*, 11(5), 264–274.

12. Rajpar, A. H., Eladwi, A., Ali, I., & Ali Bashir, M. B. (2021). Reconfigurable articulated robot. *Complexity International*, 6(6), 155.

13. Haripriya, G., Divyavani, Y., Devi, A. R., Sravan, M., & Reddy, K. (2021). Arduino based voice controlled robot. *Complexity International*. https://doi.org/10.21203/rs.3.rs-628321/v1

14. Ersahin, G., & Sedef, H. (2015). Wireless mobile robot control with tablet computer. *Procedia-Social and Behavioral Sciences*, 195, 2874–2882. https://doi.org/10.1016/j.sbspro.2015.06.411

15. Ramya, V., & Palaniappan, B. (2012). Web based embedded Robot for safety and security applications using Zigbee. *International Journal of Wireless & Mobile Networks*, 4(6), 155.

16. Meteab, W. K., Al Rikabi, H. T. S., Al Sultani, S. A. H., & Aljazaery, I. A. (2021). Controlling and monitoring a robot-car based on smart phone applications. *IOP conference series: Materials Science and Engineering*. IOP Publishing. https://doi.org/10.1088/1757-899X/1045/1/012096

17. Qadri, I., Muneer, A., & Fati, S. M. (2021). Automatic robotic scanning and inspection mechanism for mines using IoT. *IOP conference series: Materials Science and engineering*. IOP Publishing. https://doi.org/10.1088/1757-899X/1045/1/012001

18. Papcun, P., Zolotova, I., & Tafsi, K. (2016). Control and teleoperation of robot khepera via android mobile device through bluetooth and wifi. *IFAC-PapersOnLine*, 49(25), 188–193. https://doi.org/10.1016/j.ifacol.2016.12.032
29. Eren, A., & Doğan, H. (2022). Design and implementation of a cost effective vacuum cleaner robot. *Turkish Journal of Engineering*, 6(2), 166–177.
30. Kirli, A., Dilaver, M., & Çakmak, F. (2017). Mobile robot control with android device sensors by using Ros. *Mugla Journal of Science and Technology*, 3(1), 31–34. https://doi.org/10.22531/mugla.jsct.272475
31. Molnár, J., Gans, Š, & Slavko, O. (2020). Design and implementation self-balancing robot. 2020 *IEEE problems of automated electrodrive theory and practice (PAEP)*, Kremenchuk, Ukraine (pp. 1–4). IEEE. https://doi.org/10.1109/PAEP49887.2020.9240815
32. Varga, J., Vargovčík, L., & Baláž, V. (2017). Design of mobile application for controlling robosoccer via bluetooth. *Journal of Automation and Control*, 5(2), 46–49. https://doi.org/10.12691/automation-5-2-3
33. Fahmidur, R. K., Munaim, H. M., Tanvir, S. M., & Sayem, A. S. (2016). Internet controlled robot: A simple approach. 2016 *International conference on electrical, electronics, and optimization techniques (ICEEOT)* (pp. 1190–1194). IEEE.

**Publisher’s Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

**Ahmet Top** received the B.Sc. and M. Sc. degrees in electric and electronic engineering from the Firat University, Elazığ, Turkey, in 2011 and 2016, respectively. He is currently Research Assistant and working toward the Ph. D. degree at Department of Electrical and Electronics Engineering, Technology Faculty, Firat University. His current research interests include control systems and robotic.

**Muammer Gökbulut** received the B.Sc. in electric education from Gazi University in 1980 and the M.Sc. degree in electric education from Gazi University, Ankara, Turkey, in 1989, and the PhD. degree in electronic engineering from Erciyes University. He is a full Professor at Department of Electrical and Electronics Engineering, Technology Faculty, Firat University, Elazığ, Turkey. His current research interests include Neural network and Adaptive control systems.