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

Ahmet TOP (atop@firat.edu.tr)
Fırat Üniversitesi: Firat Universitesi https://orcid.org/0000-0001-6672-2119
Muammer GÖKBULUT
Fırat Üniversitesi: Firat Universitesi

Research Article

Keywords: Android application, MIT App Inventor, Bluetooth, Robot control

Posted Date: July 12th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-628321/v1

ANDROID APPLICATION DESIGN WITH MIT APP INVENTOR FOR BLUETOOTH BASED MOBILE ROBOT CONTROL

Ahmet TOP[1]*, Muammer GÖKBULUT[2]

[1*,2]Department of Electrical and Electronics Engineering, Faculty of Technology, Fırat University, Elazığ, Turkey

atop@firat.edu.tr
mgokbulut@firat.edu.tr
ORCID ID[1*]:0000-0001-6672-2119
ORCID ID[2]:0000-0003-1870-1772

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

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 [19,20] Android applications are also used for robotic systems [21-34].

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 [21]. 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 [22]. Aldhalemi et al. realized real-time control of the two-wheeled self-balancing robot by connecting with a smart phone [23]. 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 [24]. 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 [25]. 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 Microsoft Visual Basic. These are transferring camera images to the interface, detecting gas leakage and live intrusion, and manually controlling the robot. [26]. Meteab et al. carried out the control of a robot vehicle containing Arduino microcontroller and bluetooth module with Android smartphone application [27]. 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 [28]. 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 [29]. 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 [30]. 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 [31]. 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 [32]. 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 [33]. Fahmidur et al., with the Mobizen application, mirrored the screen of the Android phone to the computer and performed the robot control with bluetooth [34].

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 Figure 1 has a 4-wheel structure, each of which is connected to a 12V 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.

(a) **(b)**

Fig. 1 Mobile robot **a** front wiev; **b** rear wiev

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 Figure 2.

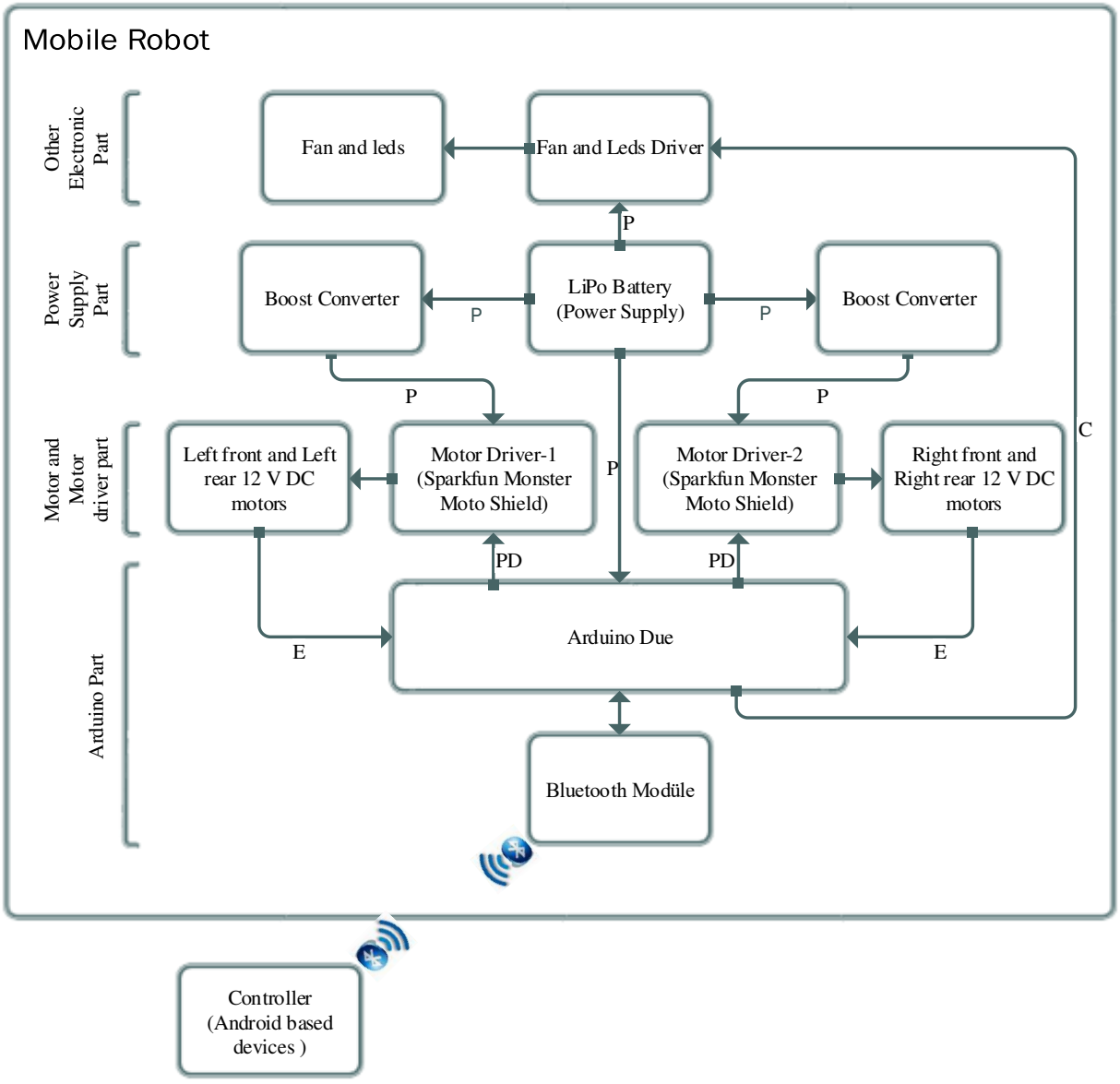


Fig. 2 Mobile Robot block diagram

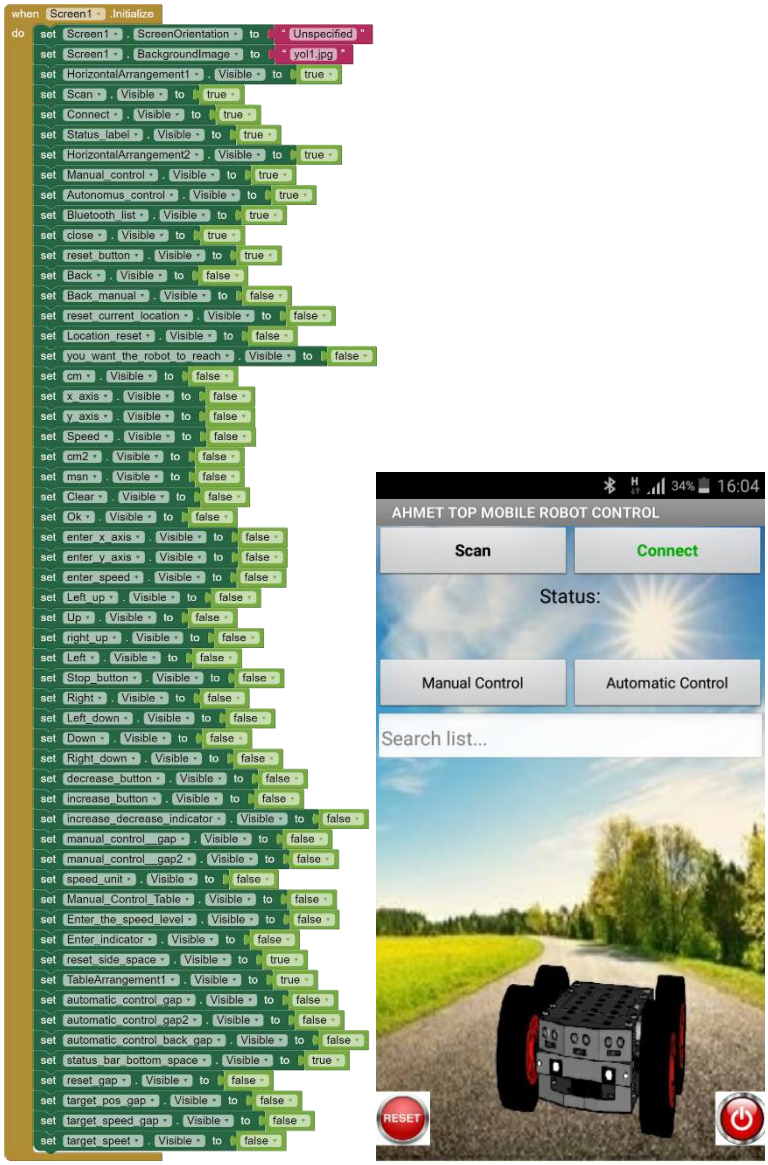
The connections between blocks in here is P:power, PD: PWM and direction E:encoder and C:Control.

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.

3.1 Main Screen

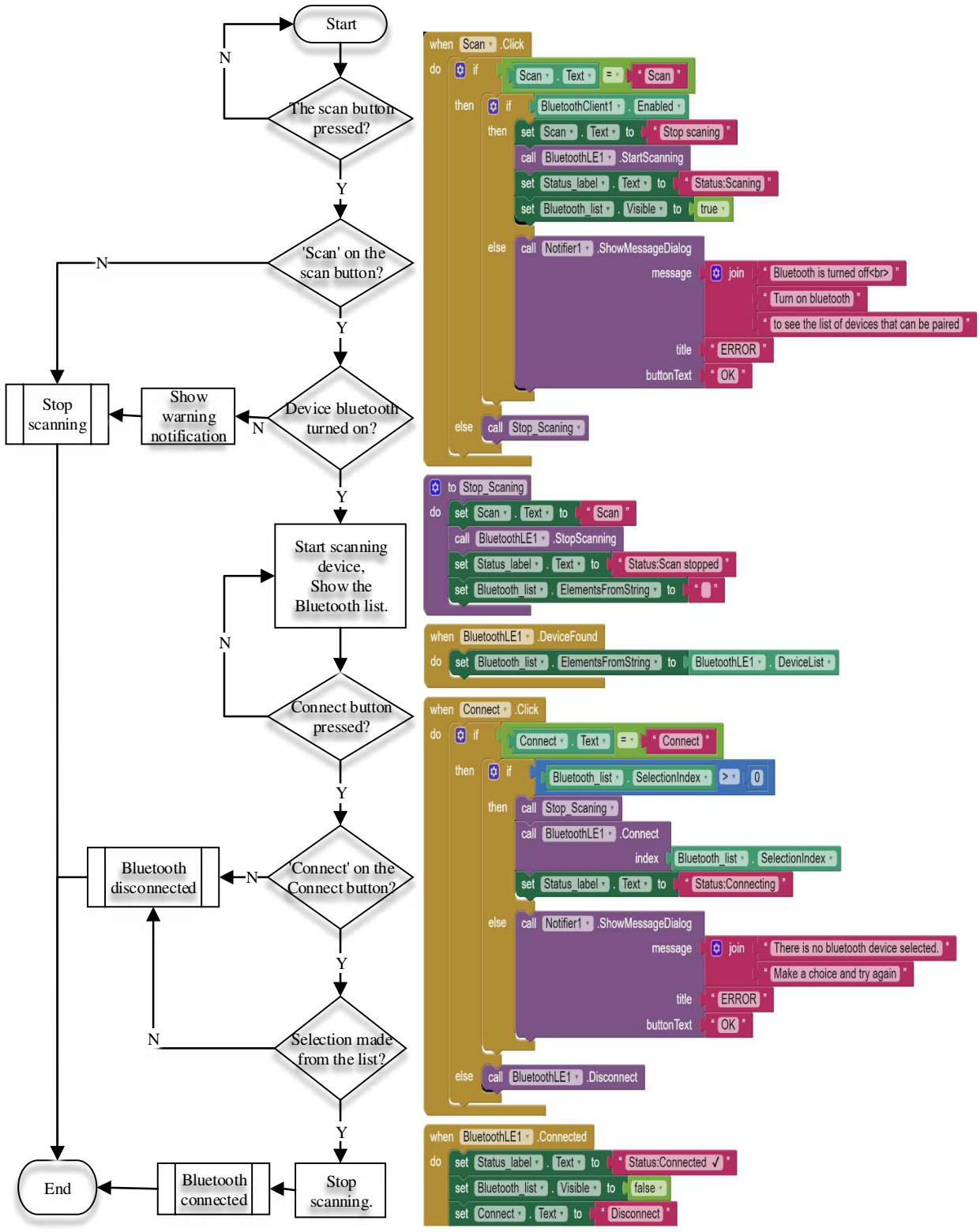
It is the screen that appears when the application is first run. On this screen, the bluetooth connection between the robot and the Android device is provided. In addition, there are selection buttons to pass to other parts, reset and application close button . Figure 3 shows the main screen interface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



(a) **(b)**

Fig. 3 a Start screen logic blocks and **b** Screenshot

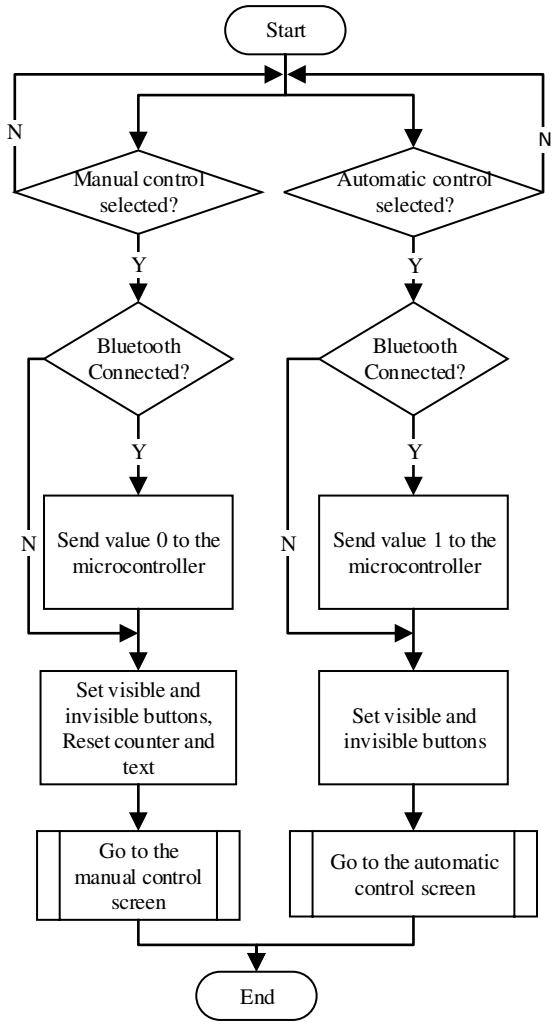
The application communicates with the robot with the bluetooth protocol. Therefore, the one needs to know the device's bluetooth address and connection status. In Figure 4, the bluetooth connection workflow diagram and the blocks that perform this flow are given.



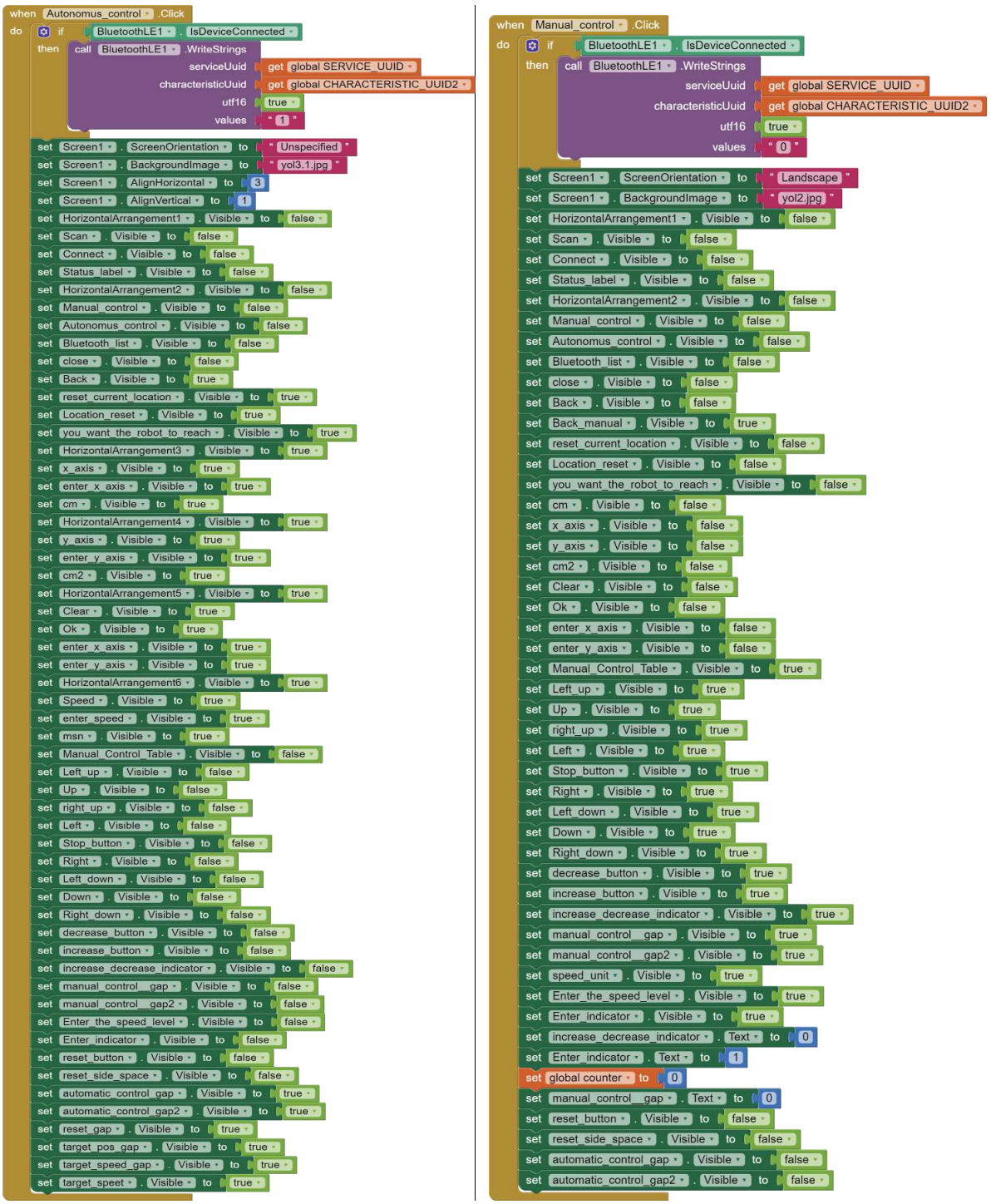
(a) **(b)**

Fig. 4 Bluetooth **a** Work flow chart and **b** logic blocks

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.



(a)



(b)

Fig. 5 Automatic control and manual control **a** work flow chart and **b** blocks

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 Figure 6.

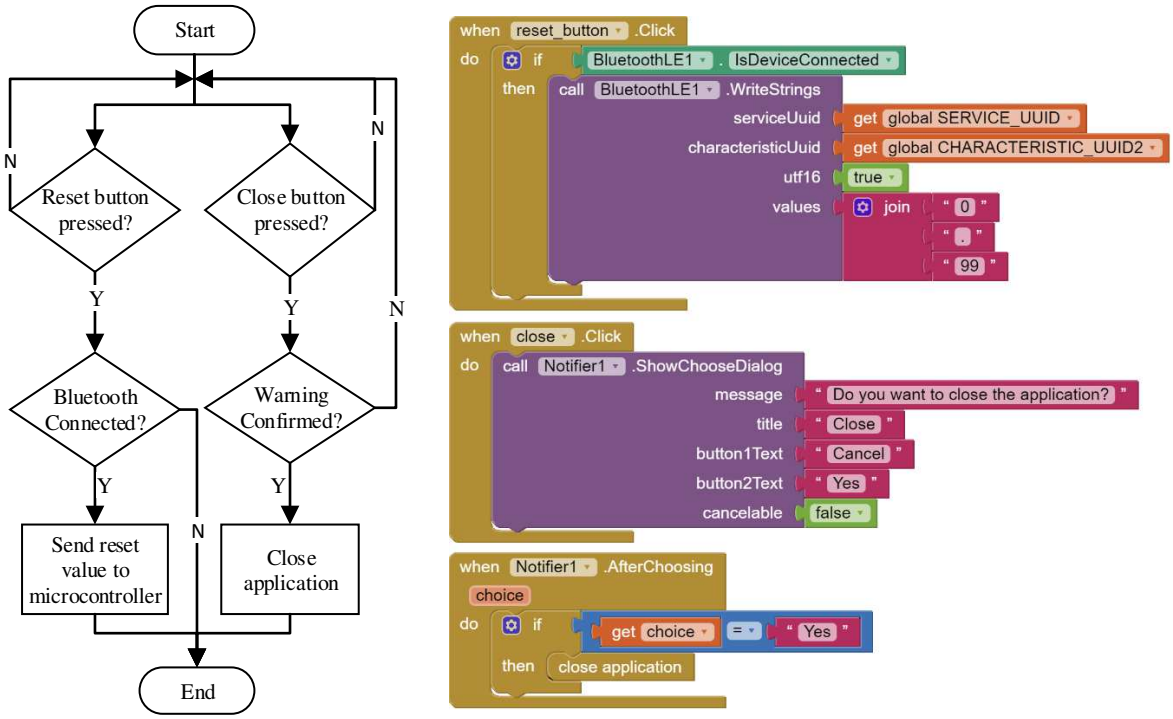


Fig. 6 a work flow chart and **b** blocks of reset and shutdown button

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 Figure 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.

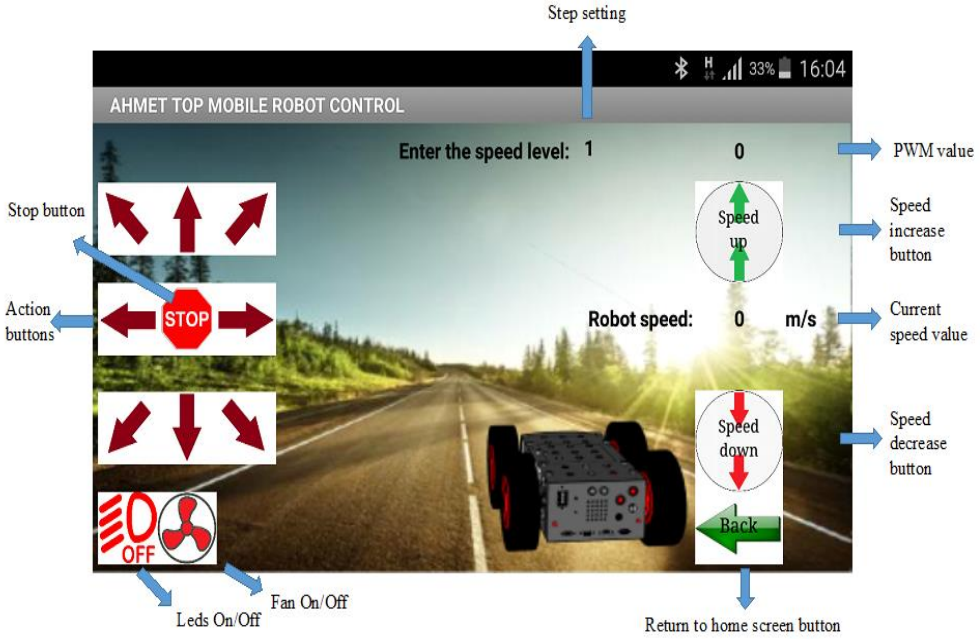
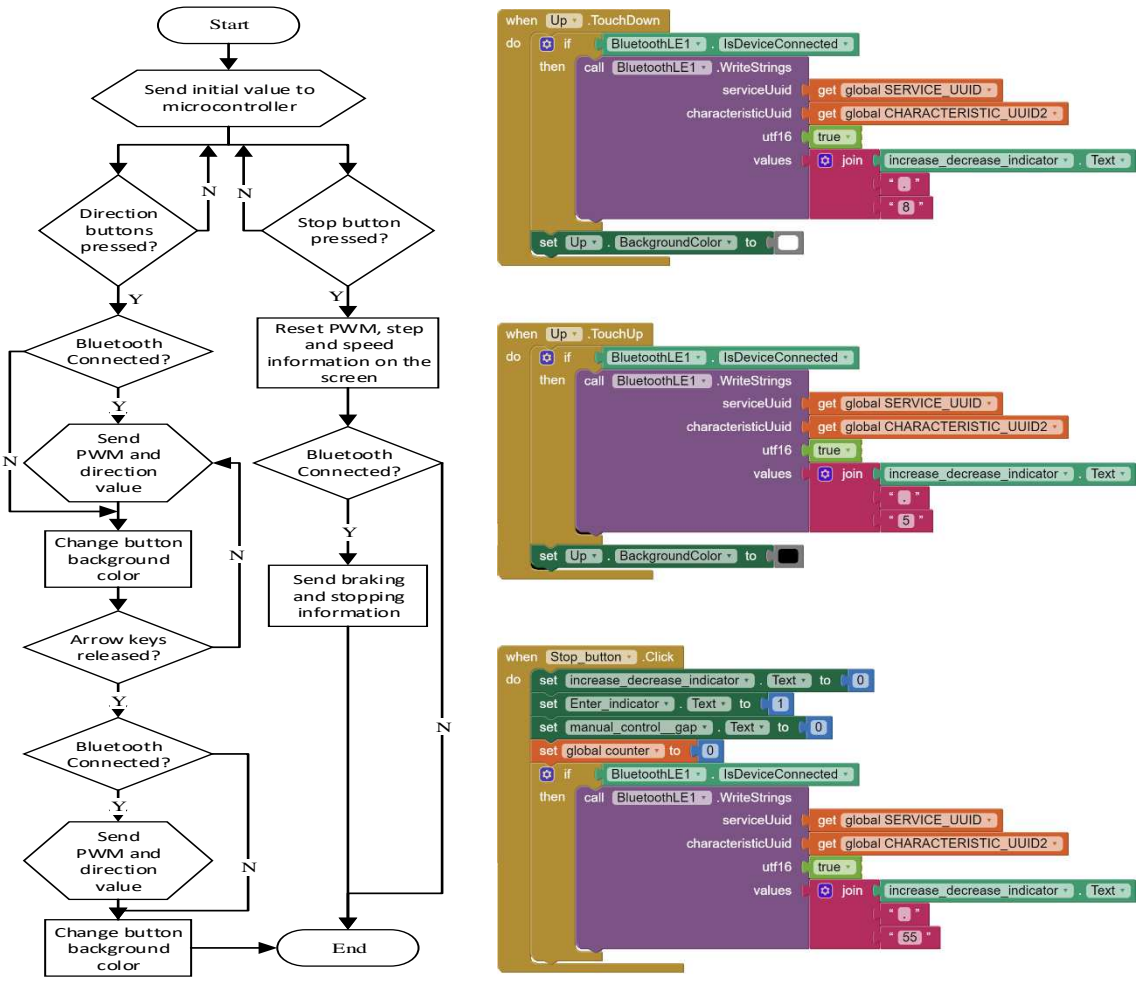


Fig. 7 Manual Control Screen

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 Figure 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.



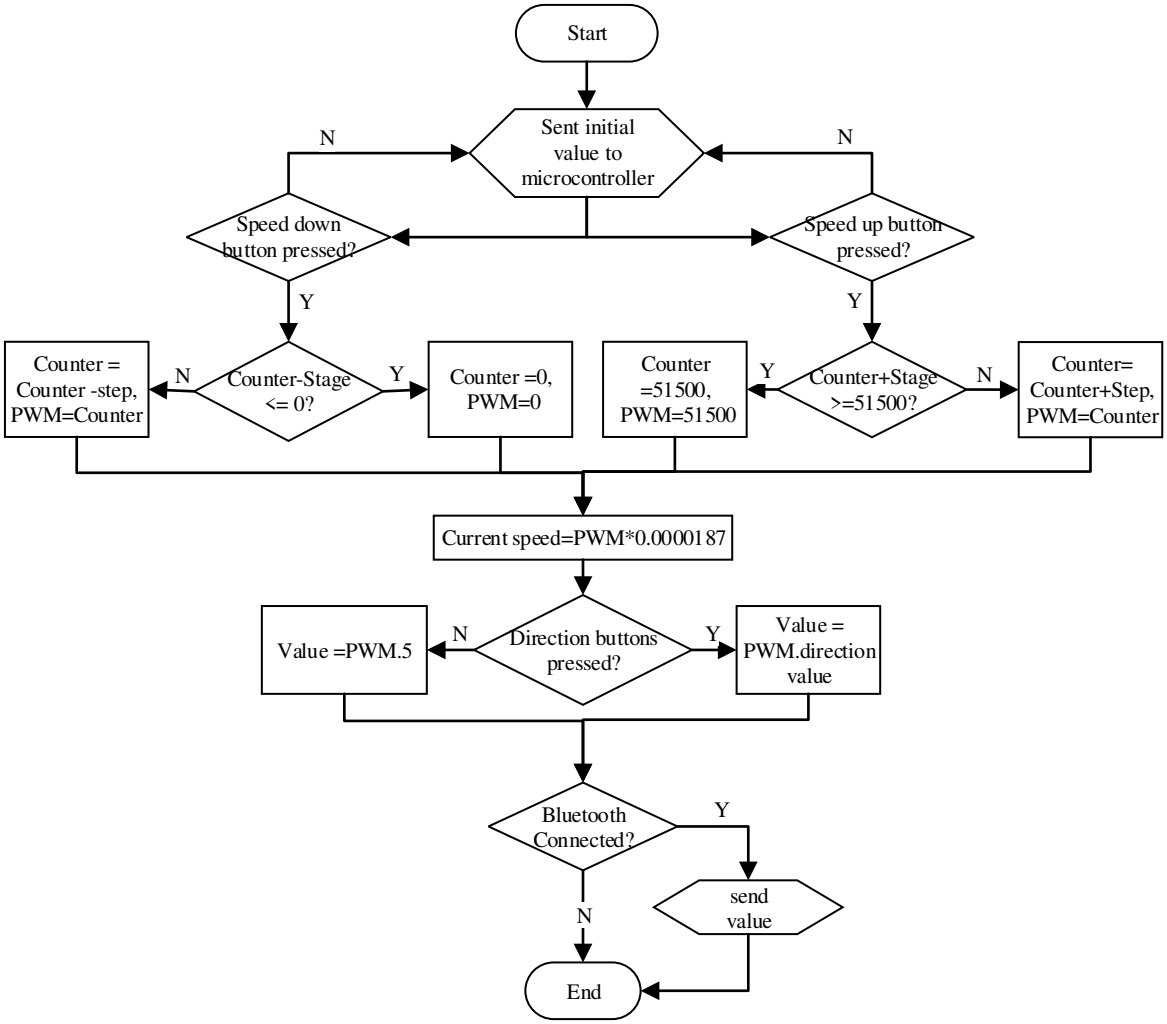
(a) **(b)**

Fig. 8 a flow chart and **b** logic blocks of direction buttons and 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 button, 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 Figure 9. Since the PWM signal is used as 16 bits in the microcontroller, it varies between 0-65500 values. However, in order to limit the motor voltage to a certain value, the PWM upper limit is chosen as 51500 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 Equation 1.

$$V=\frac{88}{60} * 2\pi r * pwm * \frac{1}{51500} \text{ m/sn} \quad (\mathbf{1})$$

where V is the motor speed, r is the wheel radius of robot and *pwm* is the entered pulse width modulation value.



(a)

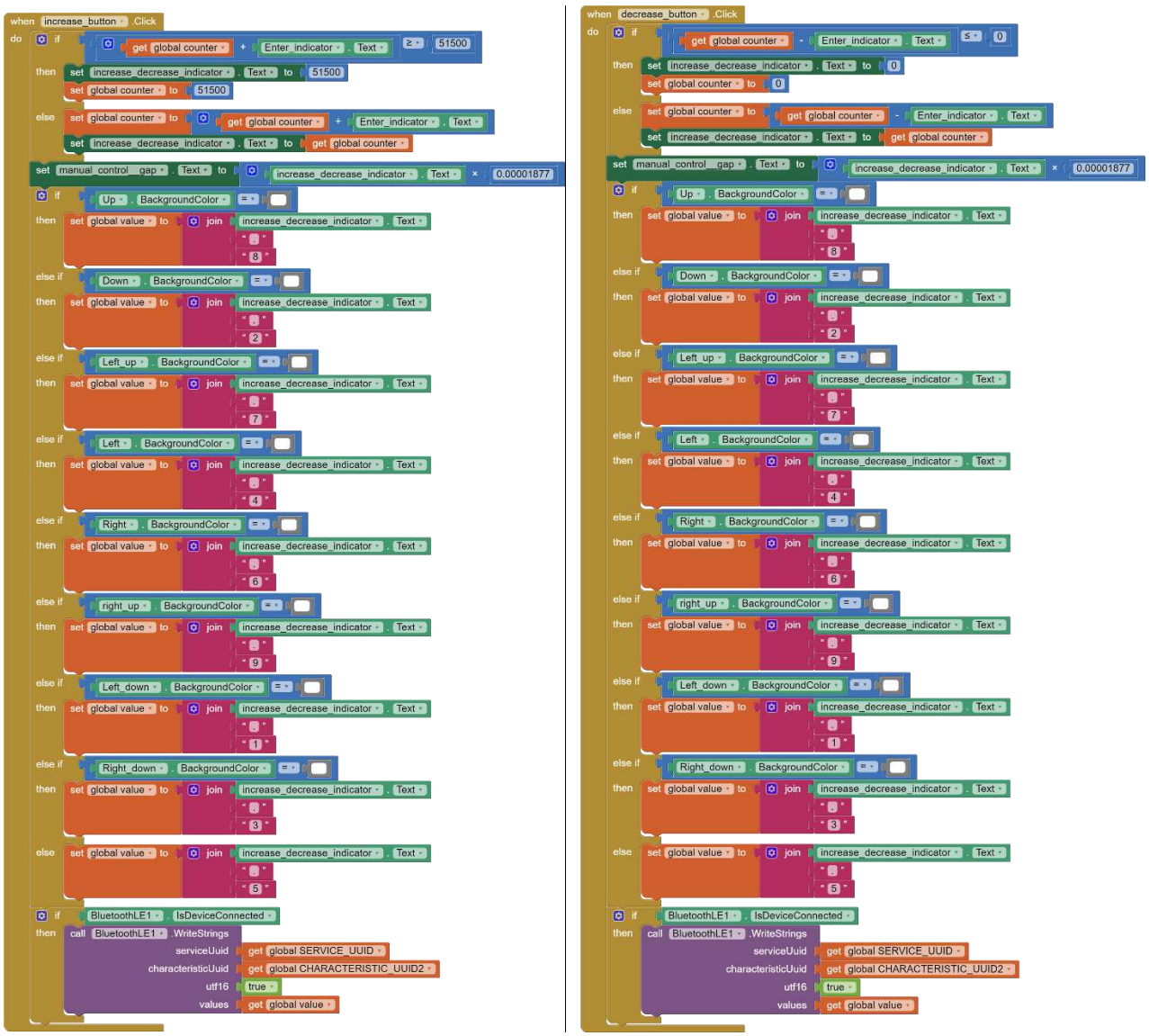
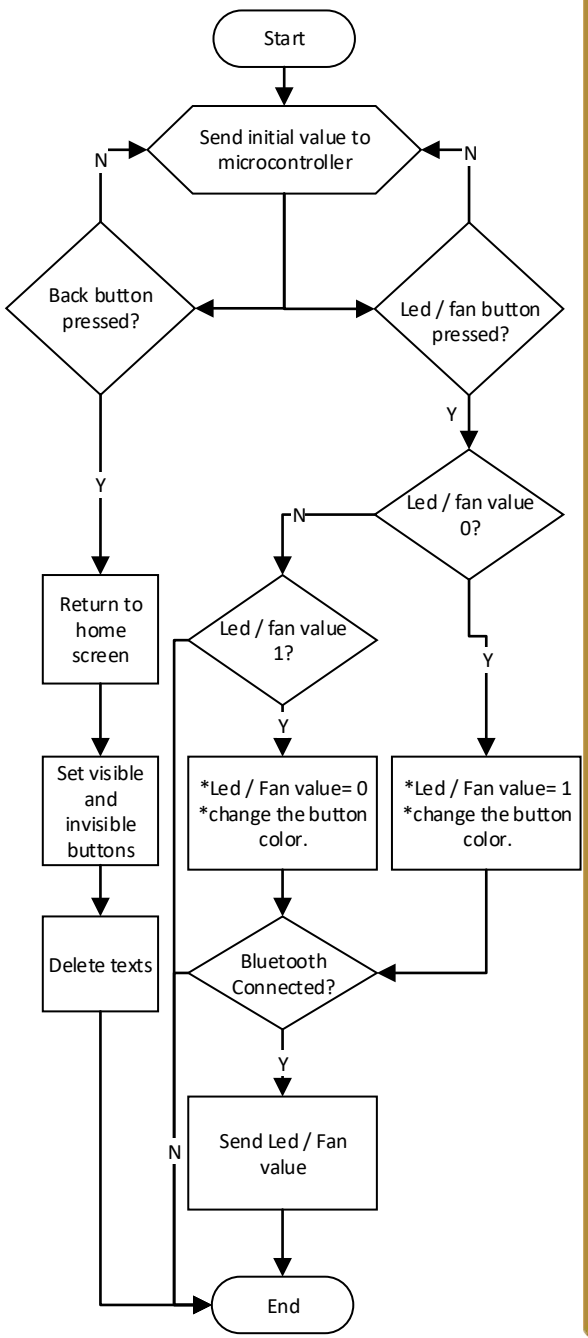


Fig. 9 a flow chart and **b** logic blocks of speed increase-decrease buttons

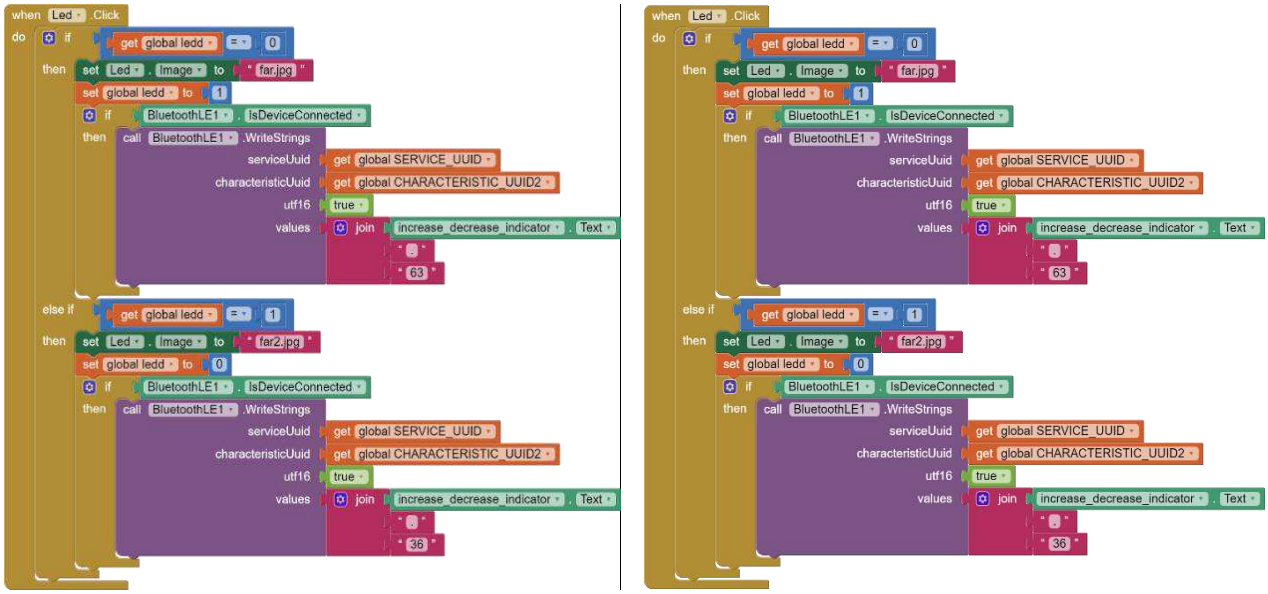
In Figure 10, flow charts and logic blocks of the return, fan and led buttons on the manual control screen are given. When the back button is pressed, the application returns to the main screen. There are four leds and one fan on the robot. While the LED button turns on and off the LEDs attached to the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


Start
Send initial value to microcontroller
N
N
Back button pressed?
Led / fan button pressed?
Y
Y
Led / fan value 0?
N
Y
Return to home screen
Led / fan value 1?
Set visible and invisible buttons
Y
*Led / Fan value= 0
*change the button color.
*Led / Fan value= 1
*change the button color.
Delete texts
Bluetooth Connected?
Y
N
Send Led / Fan value
End

(a)


when Back_manual .Click
do
set Screen1 . ScreenOrientation to " Unspecified "
set Screen1 . BackgroundImage to " yol1.jpg "
set HorizontalArrangement1 . Visible to true
set Scan . Visible to true
set Connect . Visible to true
set Status_label . Visible to true
set HorizontalArrangement2 . Visible to true
set Manual_control . Visible to true
set Autonomus_control . Visible to true
set Bluetooth_list . Visible to true
set close . Visible to true
set Back . Visible to false
set Back_manual . Visible to false
set reset_current_location . Visible to false
set Location_reset . Visible to false
set you_want_the_robot_to_reach . Visible to false
set cm . Visible to false
set x_axis . Visible to false
set y_axis . Visible to false
set cm2 . Visible to false
set Clear . Visible to false
set Ok . Visible to false
set enter_x_axis . Visible to false
set enter_y_axis . Visible to false
set enter_x_axis . Text to " "
set enter_y_axis . Text to " "
set Manual_Control_Table . Visible to false
set Left_up . Visible to false
set Up . Visible to false
set right_up . Visible to false
set Left . Visible to false
set Stop_button . Visible to false
set Right . Visible to false
set Left_down . Visible to false
set Down . Visible to false
set Right_down . Visible to false
set decrease_button . Visible to false
set increase_button . Visible to false
set increase_decrease_indicator . Visible to false
set manual_control_gap . Visible to false
set manual_control_gap2 . Visible to false
set speed_unit . Visible to false
set Enter_the_speed_level . Visible to false
set Enter_indicator . Visible to false
set reset_button . Visible to true
set reset_side_space . Visible to true
set automatic_control_gap . Visible to false
set automatic_control_gap2 . Visible to false
set Led . Image to " far2.jpg "
set global ledd to 0
set fan . Image to " fan2.jpg "
set global fann to 0
if BluetoothLE1 . IsDeviceConnected
then call BluetoothLE1 .WriteStrings
serviceUuid get global SERVICE_UUID
characteristicUuid get global CHARACTERISTIC_UUID2
utf16 true
values join " 0 "
" . "
" 99 "

(b)



(c)

Fig. 10 a flow charts of the return, fan and led buttons; **b** Back button logic blocks, **c** Fan and leds logic blocks

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 Figure 11, there are position reset button, target position and speed entry, delete and confirm button and back button on this screen.

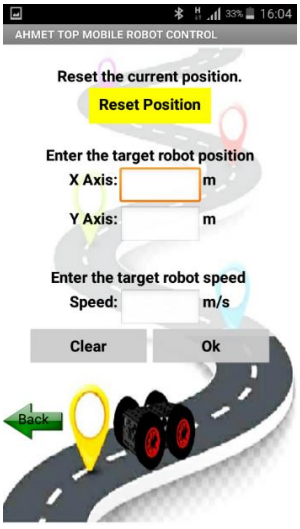
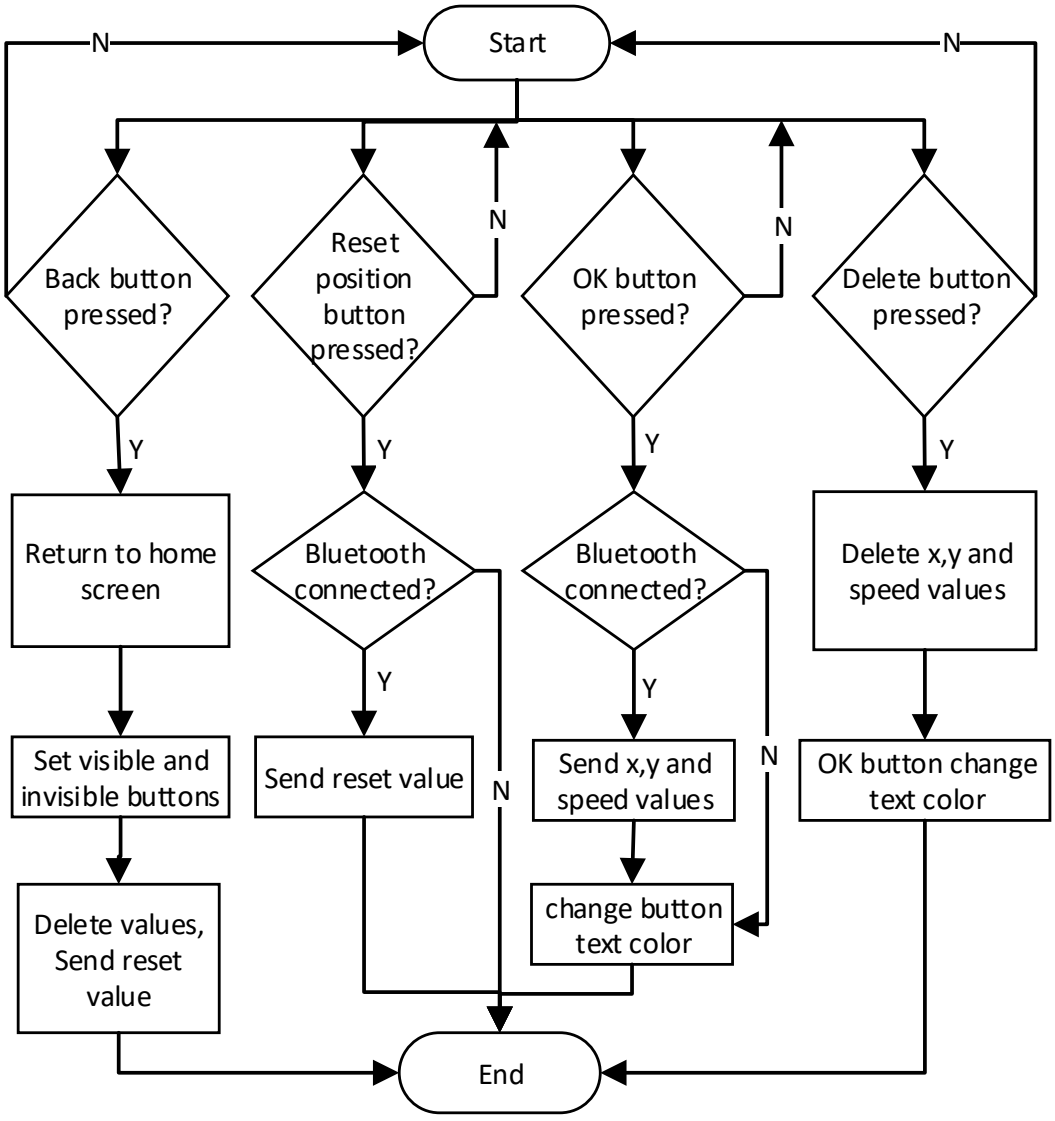
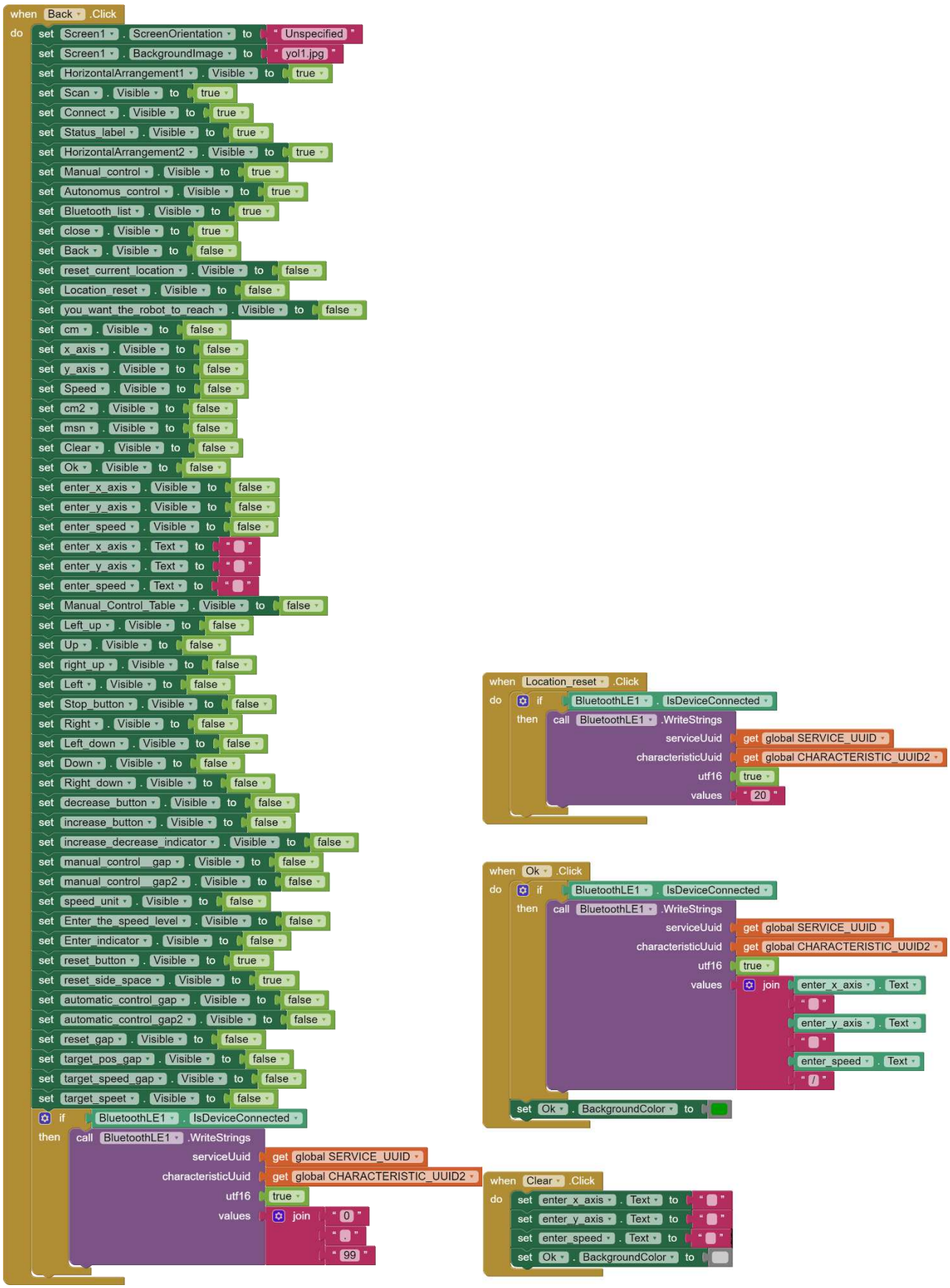


Fig. 11 Automatic control screen

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 Figure 12.



(a)



(b)

Fig. 12 a Flow chart and **b** blocks of the 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 Figure 13 was written on the Arduino Due microcontroller and a bluetooth connection was made between the smartphone and Arduino as in Figure 14.

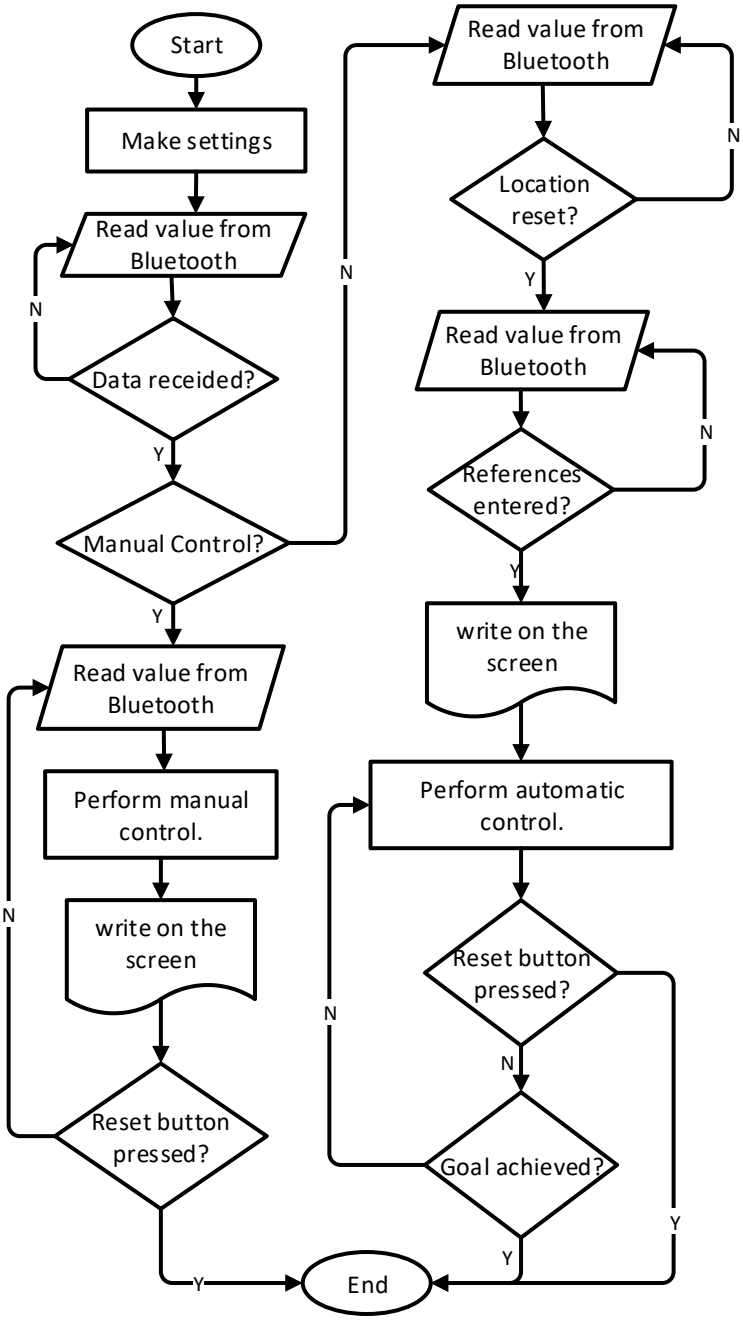


Fig. 13 Arduino Due software flow chart

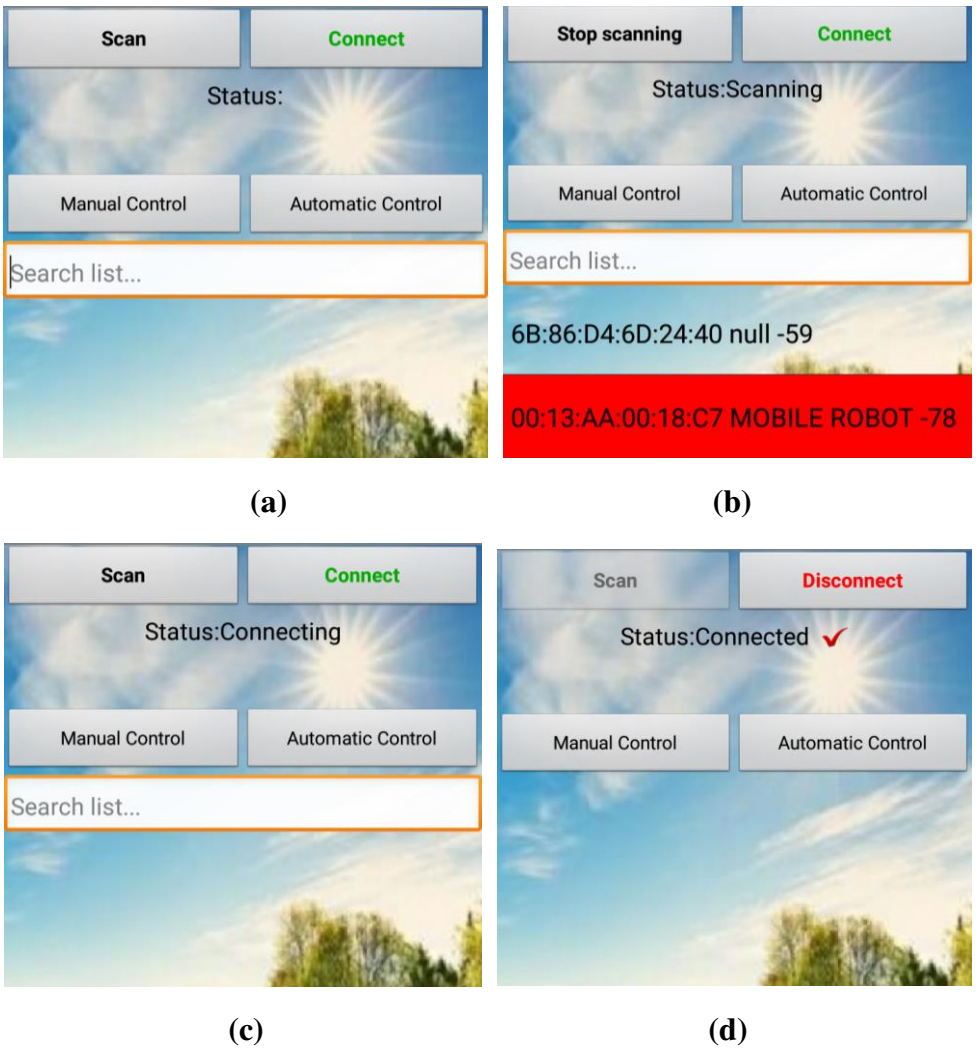
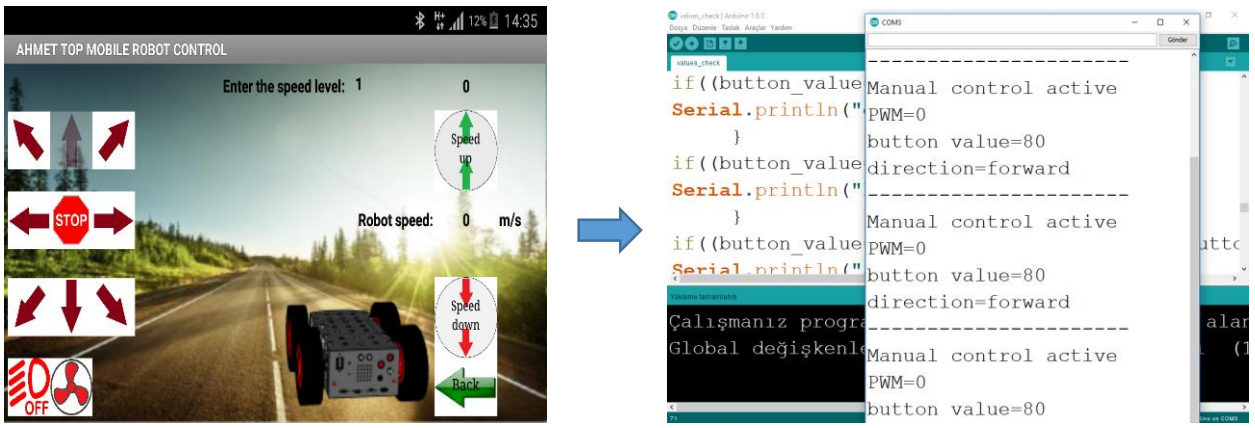
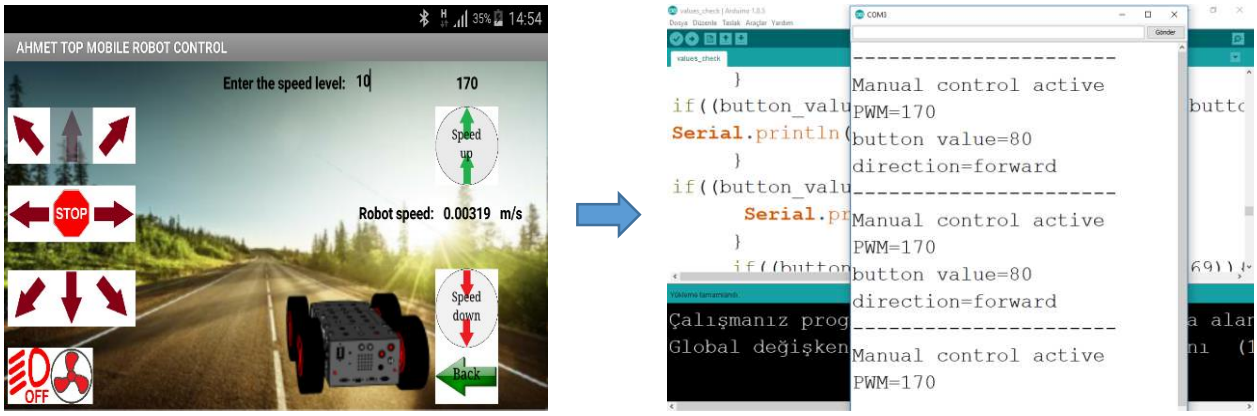


Fig. 14 Bluetooth connection: **a** Scan button is pressed; **b** Scanning and devices are listed; **c** device is selected and connect button is pressed; **d** Connected

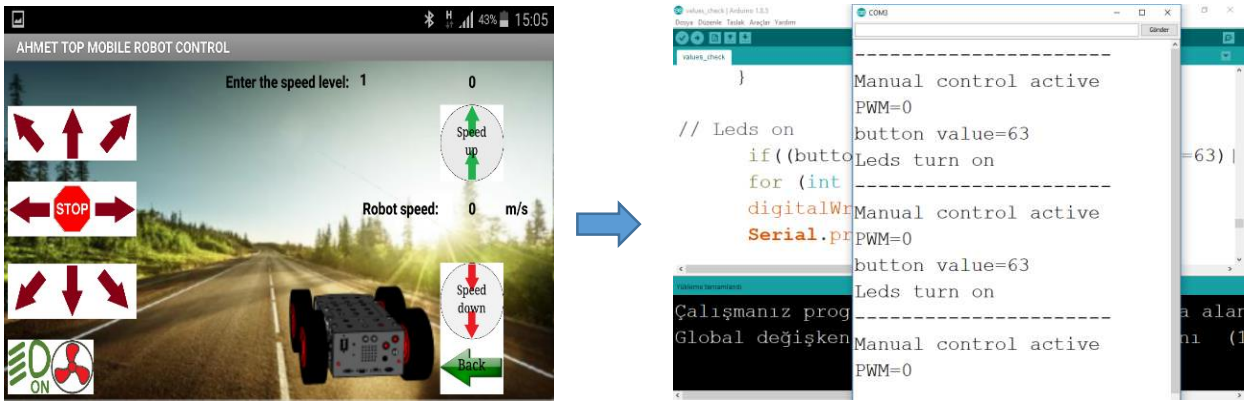
After realising a bluetooth connection between the smartphone and the Arduino microcontroller, the manual control screen in the application interface was switched and various buttons were pressed on this screen. When the buttons in the Android application are pressed, the values sent to the microcontroller 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 Figure 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.


AHMET TOP MOBILE ROBOT CONTROL
Enter the speed level: 1
0
Speed up
STOP
Robot speed: 0 m/s
Speed down
Back
OFF
Manual control active
PWM=0
button value=80
direction=forward
Manual control active
PWM=0
button value=80
direction=forward
Manual control active
PWM=0
button value=80

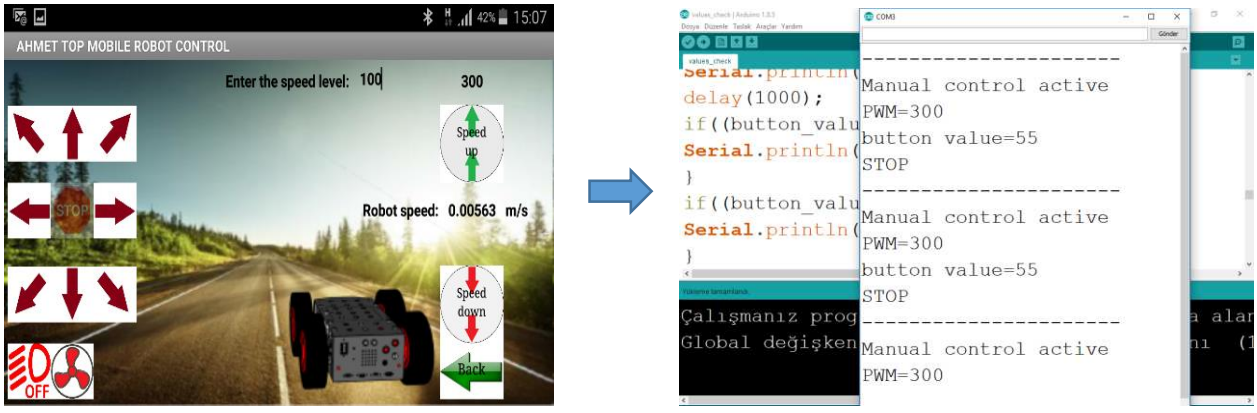
(a)


AHMET TOP MOBILE ROBOT CONTROL
Enter the speed level: 10
170
Speed up
STOP
Robot speed: 0.00319 m/s
Speed down
Back
OFF
Manual control active
PWM=170
button value=80
direction=forward
Manual control active
PWM=170
button value=80
direction=forward
Manual control active
PWM=170

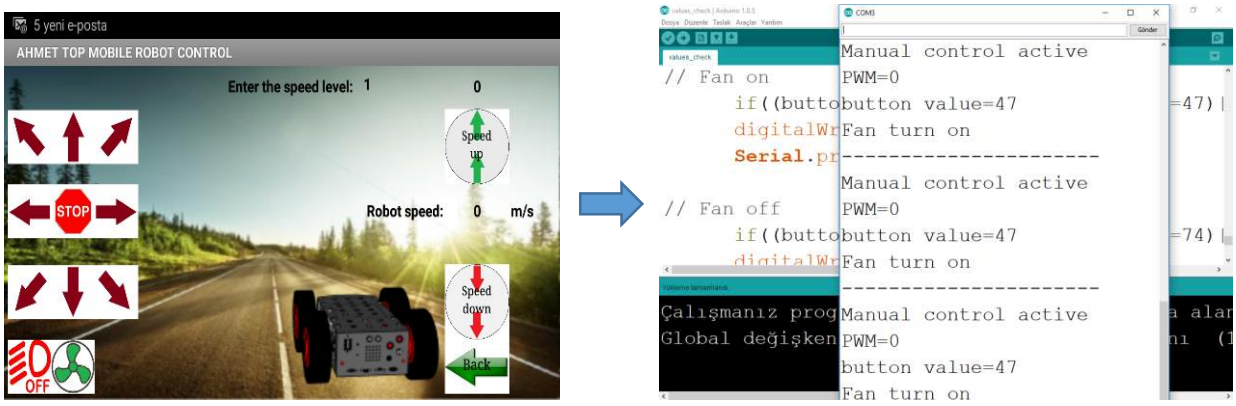
(b)


AHMET TOP MOBILE ROBOT CONTROL
Enter the speed level: 1
0
Speed up
STOP
Robot speed: 0 m/s
Speed down
Back
ON
// Leds on
Manual control active
PWM=0
button value=63
Leds turn on
Manual control active
PWM=0
button value=63
Leds turn on
Manual control active
PWM=0

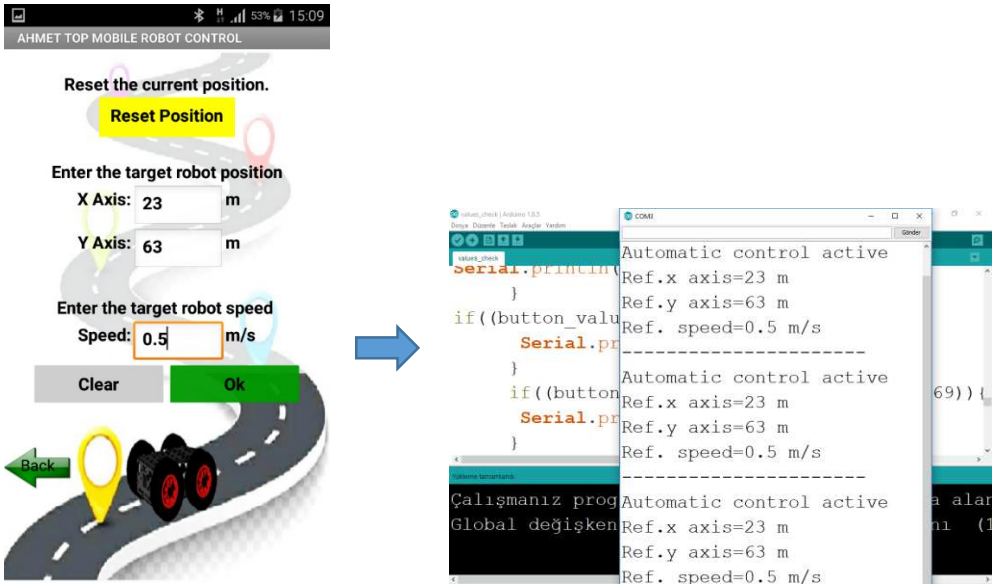
(c)


AHMET TOP MOBILE ROBOT CONTROL
Enter the speed level: 100
300
Speed up
STOP
Robot speed: 0.00563 m/s
Speed down
Back
OFF
delay(1000);
Manual control active
PWM=300
button value=55
STOP
Manual control active
PWM=300
button value=55
STOP
Manual control active
PWM=300

(d)



(e)



(f)

Fig. 14 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=0.5m/s

Figure 15 shows the speed values of each motor of the four wheel drive mobile robot in Figure 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. 20000 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 seconds, it moves in the right forward direction in 6-10.5 seconds and in the left direction in 12.5-17 seconds. 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 decrease in speeds at 4.5-10 seconds and 10.5-12 seconds. Since braking was done by pressing the stop button in 17. seconds, all four motors stopped suddenly.

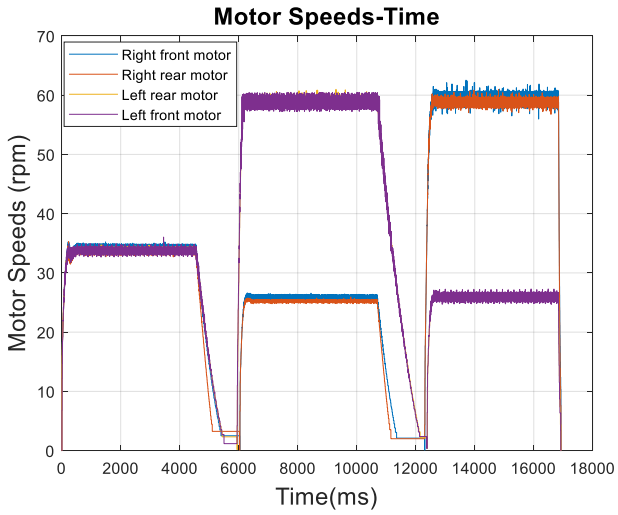


Fig. 15 Motor speed changes according to motionThe speed change graph of the robot moved in

in Figure 15, It is seen that the robot moves effectively in with the data taken from the smartphone in different directions.

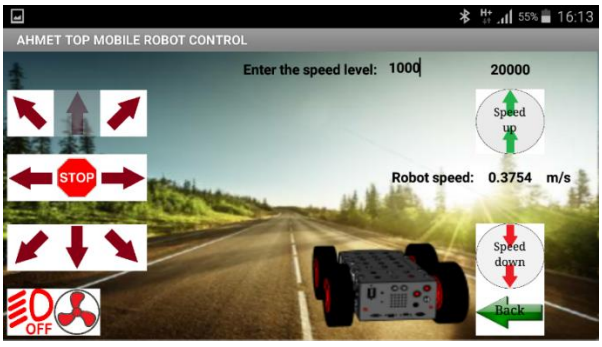


Fig. 16 Manual control screenshot

In Figure 16, the application screenshot of the movement in the 0-4.5s interval is shown. The PWM value was determined to 20000 and was sent to the microcontroller with forward button. The speed value calculated with this PWM value according to the formula in Equation 1 is 0.37 m / sec. In Figure 15, 34 rpm was measured for this time interval. When this value was converted to the unit of m / s with equation 2, it is seen that the robot speed in this range was 0.37 m / sec.

$$V = V_{rpm} * 2\pi r * \frac{1}{60} = 34 * 2\pi 0.105 * \frac{1}{60} = 0.37 \text{ m/sec} \quad (2)$$

where Vrpm is the measured speed of motors and r is the wheel radius of robot. The PID control algorithm, whose block diagram is given in Figure 17, is used to perform automatic speed and position

control of the mobile robot. As seen in Figure 18, these values were sent to the Arduino by entering the reference y-axis as 2 meters, x-axis as 0 meters and the speed as 0.22 m/s. According to Equation 3, the robot must complete this task in 9.09 seconds. When the robot position and speed graphs given in Figure 19 are examined, it is seen that the automatic movement is performed successfully.

$$t = \frac{x}{V} = \frac{2}{0.22} = 9.09\ s \tag{3}$$

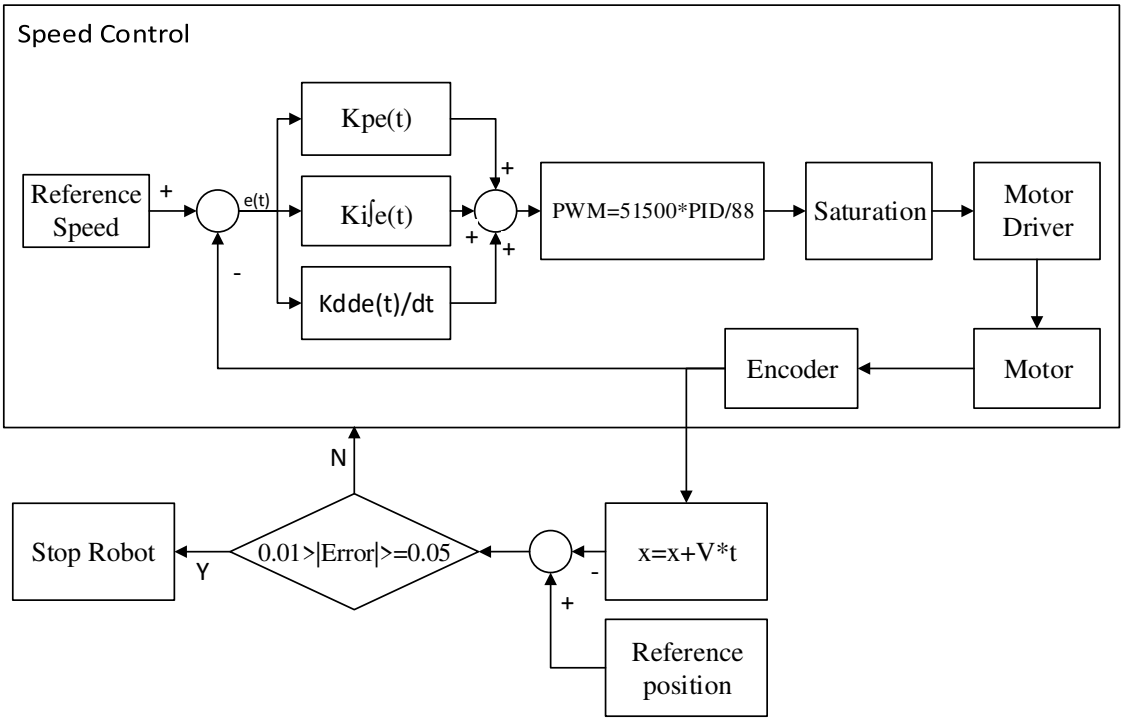


Fig. 17 Robot automatic control block diagram

In Equation 3, x is the position, V is the speed and t is the time. In the robot automatic control diagram, Kp, Ki and Kd are proportional, integral and derivative PID coefficients, respectively and e is the error signal.

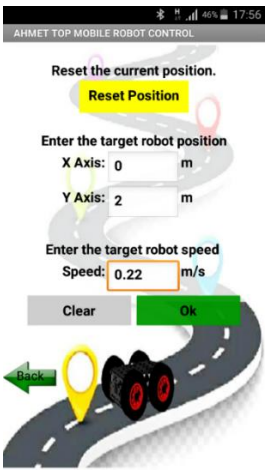


Fig. 18 References sent from the automatic control screen

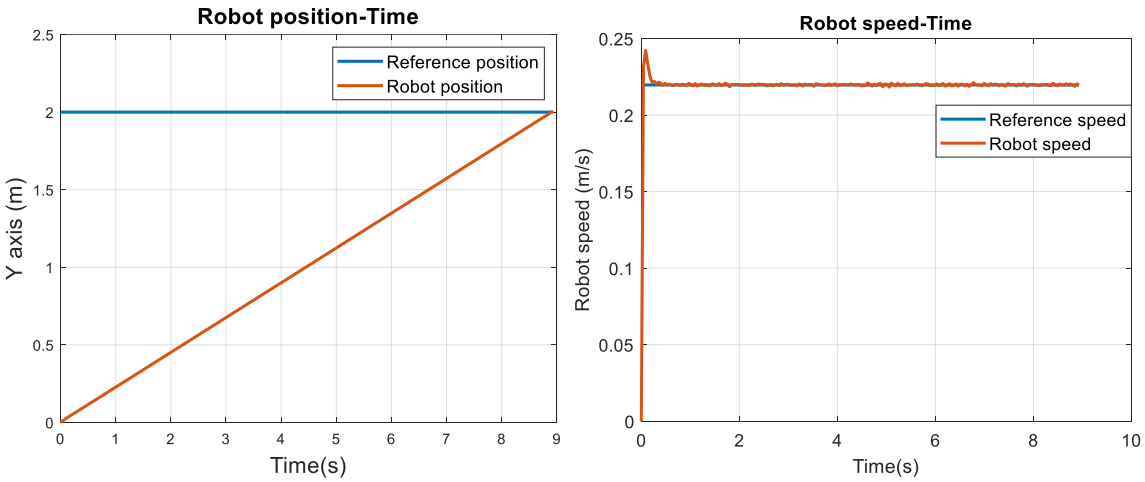


Fig 19 Mobile robot: **a** position and **b** speed control

As can be seen in Figure 19, the robot position increased linearly and the robot 2-meter reference position reached in 9 seconds with an error of 0.09 seconds. After 0.02 m/s overshoot at the start of the move, the robot was reached the steady state in 0.26 seconds and converge to the reference value.

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 Fırat University Scientific Research Projects Unit with the project number TEKF.19.07.

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

Compliance with Ethical Standards

Conficts of interest Authors declare no conficts of interest.

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

References

[1] Akilan, T., Chaudhary, S., Kumari, P., & Pandey, U. (2020, December). Surveillance Robot in Hazardous Place Using IoT Technology. In *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. https://doi.org/10.48009/4_iis_2020_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. *URL: https://code. google. com/p/blockly*, *42*.

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

[8] Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., ... & Kafai, Y. (2009). Scratch: programming for all. *Communications of the ACM*, *52*(11), 60-67.

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

[10] 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

[11] Raheem, A. K. K. A. (2017). Bluetooth Based Smart Home Automation System using Arduino UNO Microcontroller. *Al-Mansour Journal*, (27).

[12] Liu, Y., & Uthra, 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).

[13] Adiono, T., Anindya, S. F., Fuada, S., Afifah, K., & Purwanda, I. G. (2019). Efficient android software development using mit app inventor 2 for bluetooth-based smart home. *Wireless Personal Communications*, *105*(1), 233-256. https://doi.org/10.1007/s11277-018-6110-x

[14] Purkayastha, K. D., Mishra, R. K., Shil, A., & Pradhan, S. N. (2021). IoT Based Design of Air Quality Monitoring System Web Server for Android Platform. *Wireless Personal Communications*, 1-20. https://doi.org/10.1007/s11277-021-08162-3

[15] Robianto, R. (2020). Analysis Of Android Based Mobile Blocking Application Design Using Mit App Inventor. *Jurnal Ipteks Terapan*, *14*(1), 1-11. http://doi.org/10.22216/jit.2020.v14i1.5082

[16] Areed, M. F., Amasha, M. A., Abougalala, R. A., Alkhalaf, S., & Khairy, D. (2021). Developing gamification e-quizzes based on an android app: the impact of asynchronous form. *Education and Information Technologies*, 1-22. https://doi.org/10.1007/s10639-021-10469-4

[17] Asghar, M. Z., Sana, I., Nasir, K., Iqbal, H., Kundi, F. M., & Ismail, S. (2016). Quizzes: Quiz application development using Android-based MIT APP Inventor platform. *International Journal of Advanced Computer Science and Aplications*, *7*(5). https://doi.org/10.14569/IJACSA.2016.070508

[18] Singh, M., Singh, G., Singh, J., & Kumar, Y. (2021). Design and Validation of Wearable Smartphone Based Wireless Cardiac Activity Monitoring Sensor. *Wireless Personal Communications*, 1-17. https://doi.org/10.1007/s11277-021-08219-3

[19] Ahsan, M. S., Das, S., & Mobarak, H. (2020, June). Android App based Bluetooth controlled Low-cost Cloth Folding Machine. In *2020 IEEE Region 10 Symposium (TENSYMP)* (pp. 170-173). IEEE. https://doi.org/10.1109/TENSYMP50017.2020.9231012

[20] 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

[21] Rajpar, A. H., Eladwi, A., Ali, I., & Ali Bashir, M. B. (2021). Reconfigurable Articulated Robot Using Android Mobile Device. *Journal of Robotics*, *2021*. https://doi.org/10.1155/2021/6695198

[22] Nádvorník, J., & Smutný, P. (2014, May). Remote control robot using Android mobile device. In *Proceedings of the 2014 15th International Carpathian Control Conference (ICCC)* (pp. 373-378). IEEE. https://doi.org/10.1109/CarpathianCC.2014.6843630

[23] Aldhalemi, A. A., Chlaihawi, A. A., & Al-Ghanimi, A. (2021, February). Design and Implementation of a Remotely Controlled Two-Wheel Self-Balancing Robot. In *IOP Conference Series: Materials Science and Engineering* (Vol. 1067, No. 1, p. 012132). IOP Publishing.

[24] Harıprıya, G., Dıvyavanı, Y., Devı, A. R., Sravan, M., & Reddy, K. (2021). Arduıno Based Voıce Controlled Robot. Complexity International, 25(01).

[25] 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

[26] 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.

[27] Meteab, W. K., ALRikabi, H. T. S., Al Sultani, S. A. H., & Aljazaery, I. A. (2021, February). Controlling and Monitoring a Robot-Car Based on Smart Phone Applications. In IOP Conference Series: Materials Science and Engineering (Vol. 1094, No. 1, p. 012096). IOP Publishing. https://doi:10.1088/1757-899X/1094/1/012096

[28] Qadri, I., Muneer, A., & Fati, S. M. (2021, February). Automatic robotic scanning and inspection mechanism for mines using IoT. In *IOP Conference Series: Materials Science and Engineering* (Vol. 1045, No. 1, p. 012001). IOP Publishing. . https://doi:10.1088/1757-899X/1045/1/012001

[29] 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

[30] Eren, A , Doğan, H . (2022). Design and implementation of a cost effective vacuum cleaner robot. Turkish Journal of Engineering , 6 (2) , 166-177 https://doi.org/10.31127/tuje.830282

[31] Kırlı A., Dilaver M., Çakmak F., (2017), Mobıle Robot Control Wıth Androıd Devıce Sensors By Usıng Ros, Mugla Journal of Science and Technology, Vol 3, No 1, Pages 31-34 https://doi:10.22531/muglajsci.272475

[32] Molnár J., Gans Š. and Slavko O.,(2020), "Design and Implementation Self-Balancing Robot," 2020 IEEE Problems of Automated Electrodrive. Theory and Practice (PAEP), Kremenchuk, Ukraine, pp. 1-4, https://doi:10.1109/PAEP49887.2020.9240815.

[33] 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: 10.12691/automation-5-2-3.

[34] Fahmidur, R. K., Munaim, H. M., Tanvir, S. M., & Sayem, A. S. (2016, March). Internet controlled robot: A simple approach. In *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)* (pp. 1190-1194). IEEE.

Ahmet TOP received the B.Sc. and M. Sc. degrees in electric and electronic engineering from the Fırat University, Elaziğ, 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.