Remote Control Robot Using Android Mobile Device

Jan Nádvorník
Department of Control Systems and Instrumentation
VSB – Technical University Ostrava
Ostrava, Czech Republic
jan.nadvornik@centrum.cz

Pavel Smutný
Department of Control Systems and Instrumentation
VSB – Technical University Ostrava
Ostrava, Czech Republic
pavel.smutny@vsb.cz

***Abstract* — The paper describes the design and realization of the mobile application for the Android operating system which is focused on manual control of mobile robot using wireless Bluetooth technology. The application allows the robot control interaction with the display, or voice. When we use a graphical interface, we can monitor the current distance of the robot from obstacles. The measurement of distance is carried out by ultrasonic sensor placed in front of the robot. It was necessary to build a prototype of a mobile robot for the development of the application. The prototype of the mobile robot is based on the differential gear.**

Keywords—mobile application; Android; mobile robot

I. INTRODUCTION

Exponential growth in the mobile device market over the last several years changed devices from telephone conversations into small pocket personal computers with operating system. They also expanded from touch driven devices to voice command devices. They are not only using GSM/UMTS cellular phone networks, but also Bluetooth and Wi-Fi wireless technology standard for exchanging data with another an electronic device. All this possibilities created another ecosystem and manufacturers of various construction kits are using mobile devices as remote control. Construction kits contain servo motors, touch, distance or light sensors. Many kinds of real-life embedded systems, from elevator controllers to industrial robots, may be modelled using these construction kits. [1]

Design and realization of the mobile application for the Android operating system and mobile robot was realised for the purpose of engineering education. Mobile application for Android operating system was developed in Eclipse integrated development environment and for mobile robot the Lego Mindstorms construction kit was used.

II. MOBILE APPLICATION DEVELOPMENT

Mobile application development is the process by which application software is developed for low-power handheld devices, such as mobile phones or tables. Mobile app development has been steadily growing, both in terms of revenues and jobs created [2].

Android software development is the process by which new applications are created for the Android operating system. Applications are usually developed in the Java programming language using the Android Software Development Kit, but other development tools are available. The officially supported integrated development environment (IDE) is Eclipse using the Android Development Tools (ADT) Plugin. Eclipse is an IDE which contains a base workspace and an extensible plug-in system for customizing the environment. Written mostly in Java, Eclipse can be used to develop applications in Java. By means of various plugins, Eclipse may also be used to develop applications in other programming languages [2].

The Android software development kit (SDK) includes a comprehensive set of development tools. These include a debugger, libraries, a handset emulator based, documentation, sample code, and tutorials. A mobile device emulator is a virtual mobile device that runs on computer and lets programmer to develop and test Android applications without using a physical device. The Android emulator mimics all of the hardware and software features of a typical mobile device, except that it cannot place actual phone calls, there is no support for Bluetooth and determining network connected state.

III. BUILDING A MOBILE ROBOT

Controlling a robot with a mobile device like smartphone or a tablet that runs Android operating system must have Bluetooth module. The robot has to be built before to be controlled while the Bluetooth module has to be included in the project. Robotic parts for the model included structure (for example car chassis), the controller, Bluetooth module, electric motors, motor driver, and other parts like batteries, power cables, wheels, etc. It is not enough to have an Android device and a robot. These two devices have to be connected and programmed to share information. On the Android side has to be available an application with interface to be used by the user to send commands to the robot.

The Lego Mindstorms construction kit was used to build a mobile robot. It contains software and hardware to create customizable, programmable robots. It includes a programmable brick computer that controls the system, a set of modular sensors and motors, and Lego parts from the Technics line to create the mechanical systems. The Mindstorms Robotics Invention System kit contained three servo motors and one light, sound and distance as well as 2 touch sensors and support for 4 without using a sensor multiplexer. The Lego Mindstorms programming is command box programming, rather than code programming. Very

978-1-4799-3528-4/14/$31.00 ©2014 IEEE

simple programs can be created using the menu on the NXT Intelligent Brick. The kit includes also software for writing programs. NXT-G is a graphical programming environment that comes bundled with the NXT. With careful construction of blocks and wires to encapsulate complexity, NXT-G can be used for real-world programming, but for the mobile robot was used the Bricx Command Center (BricxCC). It is the integrated development environment (IDE) of Next Byte Codes (NBC), Not Quite C (NQC) and Not eXactly C (NXC) and have a syntax like C.

A. NXT Intelligent Brick

The main component in the construction kit is brick-shaped computer with a 32-bit microcontroller, Bluetooth communications and USB port. It can take input from up to four sensors and control up to three motors. The brick has a 100×60 pixel monochrome LCD display, speaker and can play sound files at sampling rates up to 8 kHz and four buttons that can be used to navigate a user interface using hierarchical menus [2].

The flash memory is used not only for firmware, but also for saving programs, pictures and sounds. The flash memory is rewritable. The firmware for the NXT intelligent brick provides the control program for the device and has been released as open source, along with schematics for all hardware components. One of the task of the firmware is to cooperate with the programing language. Different firmware can be flashed to the NXT Intelligent Brick. [4; 5]

B. The Servo Motors

The movement of the robot is provided by the servo motors. Interactive servo motor consists of tachometer for built-in rotation sensor, motor core, built-in gearing and hub with an axle hole for attaching a wheel.

The servo motor has a built-in rotation sensor that measures speed and distance, and reports back to the NXT intelligent brick. The rotation sensor measures motor rotation in degrees or full rotations (accuracy of +/- one degree). One rotation is equal to 360 degrees. Several motors can be aligned to drive at the same speed [4].

C. Ultrasonic Sensor

The ultrasonic sensor can measure the distance from the sensor to something that it is facing, and detect movement. It can show the distance in both centimetres and inches. The maximum distance it can measure is 233 cm with a precision of 3 centimetres. The ultrasonic sensor works by sending out ultrasonic sound waves that bounce off an object ahead of it and then back. It senses the time it took for that to happen. This sensor is only accurate at detecting flat surfaces – it can detect object within angle at maximum 15 degrees [6].

D. Robotic Model Building

With the knowledge of the Lego Mindstorms construction kit capabilities, the mobile robot was built. The first decision was connected with the type of the chassis. One option was to use differential chassis, other to use chassis similar to cars. The first prototype was based on the car chassis, but the accuracy towards turning the model was very low. Therefore second version of the model is based on the differential chassis (Fig. 1).

Fig. 1 A mobile robot

The movement of the robot is provided by the servo motors each of them controls one caterpillar track. There is NXT intelligent brick above them and the ultrasonic sensor in the front. To operate with the model with highest accuracy, the number of the cogwheels was minimised. One of the motor receives every 25 ms information about the speed and angle of the rotation. The rotation of the model is based by the slowing down one of the motors, for the direct orientation the speed values are same for both motors. For the rotation of the model on the spot speed values are also same for both motors, but with opposite orientation.

IV. TEST APPLICATION FOR MODEL CONTROL

After creating a model of the mobile robot was another part to design and implement a mobile application. First, it was necessary to put the wireless communication between a model and a mobile device into operation. For this purpose a test application in C# was created in which has been proposed the user interface. After successful testing of C# application, the code has been implemented in the Java environment.

A. Communication Between the Model and the Mobile Device

For communication between the model and the mobile device, we can use Bluetooth or Wi-Fi. Regarding the transmission speed, communication via Bluetooth, which in asynchronous mode can handle 720 kbps is compared to Wi-Fi with 11Mbps several orders slower. Wi-Fi has also a greater reach, but for the purpose of this work it was not important. On the other hand, because Bluetooth has lower power requirements than Wi-Fi, for this reason is much smaller and is thus part of NXT cube. Therefore, the external Wi-Fi module is not included in the kit. For the purpose of this work

the communication between the mobile device and the robot the Bluetooth technology was used.

Communication on the NXT units was created in the development environment Bricx, based on the syntax of C language. Before beginning any communication, it is necessary to check whether the Bluetooth device in NXT and mobile device are connected to each other. This is ensured by function *BTCommCheck*. If you run a program in NXT and this function does not return *NO_ERR*, the connection is not established. The status of the NXT then display "Bluetooth not connected" and device also emits an intermittent tone. If the connection is established, the device is ready to receive (*ReceiveMessage*), respectively to send data (*BTSendMessage*).

For sending and receiving data from an external device we have to use following format. The first field (*__buffer[0]*) determines whether a response is required. In this case, the value 0x80 means that it is not. The following value 0x09 is a write command. Field *__buffer[2]* is called a mailbox. In the mailbox are variables such as speed and distance. The last field is stored the total message size. All constants and variables in the string must be byte type.

B. The Test Application in C#

The main reason for creating test applications in C# was the put communication between the PC and the NXT into operation. Since this part was time consuming, it was therefore appropriate to test communication in Java environment, where is application created for PC, compiled and loaded into the mobile device. For this reason PC with built-in Bluetooth device was used.

Test application (Fig. 2) includes only a field for serial port selection, button for connect/disconnect from the device and text box for received a string of NXT. Once communication from NXT to PC was operational, to the test application was added a text box and a button to send a string of characters that was inscribed into this field. String received by the NXT device was subsequently sent using TextOut on the display.

After launching two-way communication NXT unit was connected to servomotor. For C# was created a slider that is coupled to a variable *speed*. The range was set to 0-255. In the Bricx a function *OnFwd*(*OUT_B*, *speed*) was used for receiving data from the PC. This function has two parameters. The first parameter is for setting the output port to which the motor is connected. In this case port B. Another parameter is the speed which user sets by using the slider (variable *speed*).

Function *OnFwd* works by turning the engine on the level at the desired speed. It was experimentally found that the values 0-128 are designed to rotate the motor forward and values 129-255 for reverse direction. In the edge points, it means 0 and 255 is motor at the rest. However, the mobile model is due to the structure of the robot movement direction opposite to the sense of rotation. For values of 255 to 512 the robot works analogously.

Further, in the environment of the C# language was created a proposal application screen through which the robot will be controlled (Fig. 2). It is a trackball which is after the start of the application located in the middle of the screen. In this position the robot receives *speed* = 255 so the robot stops. The maximum speed in the forward direction is achieved if the trackball along the vertical axis drawn up to the upper edge of the screen. At this time *speed* = 128. For maximum speed in the backward direction is a speed value *speed* = 383 and trackball position on the screen is on the vertical axis at the bottom edge. Turning the robot was in test mode solved by stopping one of the motors.

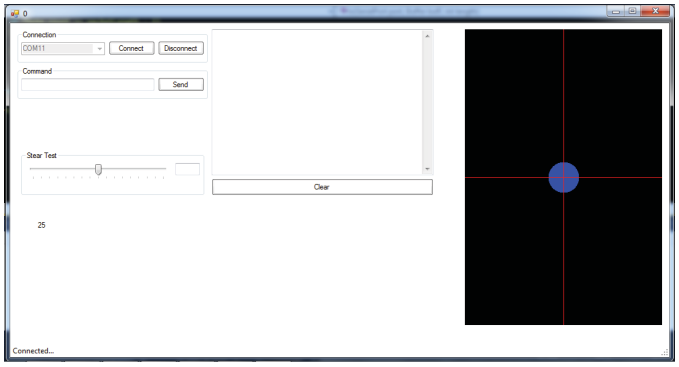

Fig. 2 Test application

V. MOBILE APPLICATION

In previous research [7] the App Inventor development environment was used to develop a mobile application to remotely control the robot which is built from Lego Mindstorms NXT kit. The App Inventor is visual development software for the Android platform. It is an open-source web application originally provided by Google, and now maintained by the Massachusetts Institute of Technology (MIT).

There are other mobile applications for remote control of the NXT device available in a digital distribution platform for applications for the Android operating system (Google Play). Developers took different approaches for the user interface – some applications use joystick, arrows or accelerometer to control the NXT device.

A. Mobile Application in a Java Language

After creating a test application for control the robot had to be all communication between the PC and the NXT device rewritten to Java. On the basis of the proposal was created user interface, which was complemented by a menu and the bar at the top of the screen. After connecting to the NXT device, model of the robot can be controlled with the display or via voice input. For the voice control is necessary to connect to the Internet. The application was optimised for the version of Android 4.0.3, for which the voice recognition takes place online.

B. User Interface

The status of the Bluetooth adapter is first check after launching the application. If it is turned off, the application asks the user for permission to turn it on. If you select “Yes”, the Bluetooth will turn on together with an application's user interface. In the case that the switching is not permitted, the application is terminated. If Bluetooth is turned on before

starting the application itself, this call is not displayed and the application will start immediately.

At the top of the screen (Fig. 3) is the title of the application and the information that the device is not connected. This status also indicates the red-colored trackball located in the middle of the screen. It is also not possible to move with him. It is thus necessary to invoke the context menu, select *Connect*, ant then select the device to be connected. If the device is not in the list, you must select option *Scan for devices* and perform pairing. If the pairing is successful, the next time when you connect to any of the devices which are in the list of paired devices connection will be automatically activated.

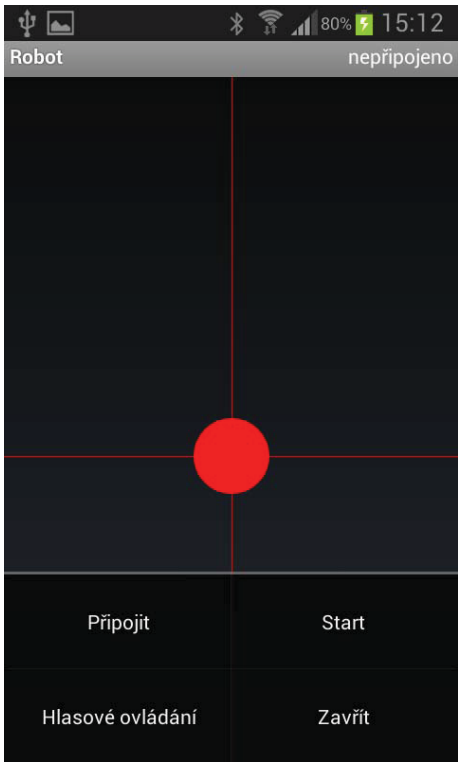

Fig. 3 User interface

After a successful connection to the robot, this state is displayed in the upper bar of the application and trackball colour is now orange. If the program is running in a robot, it is possible in a mobile application using the *Start* button to start a thread for receiving and thread for sending data from your mobile device. This state is indicated by the green colour of the trackball. In the top bar shows the distance of the robot from the barrier (Fig. 4). When you touch the screen - trackball turns blue, indicating outgoing communication and robot moves in the selected direction. In the case, that the contact with the screen is interrupted, trackball will returns to the original position and the robot will stop. The application can be terminated using the *Close* button in the menu. In a case that the connection via Bluetooth was interrupted, the mobile application displays on the display „Connecting to device has been lost“ and application will be terminated.

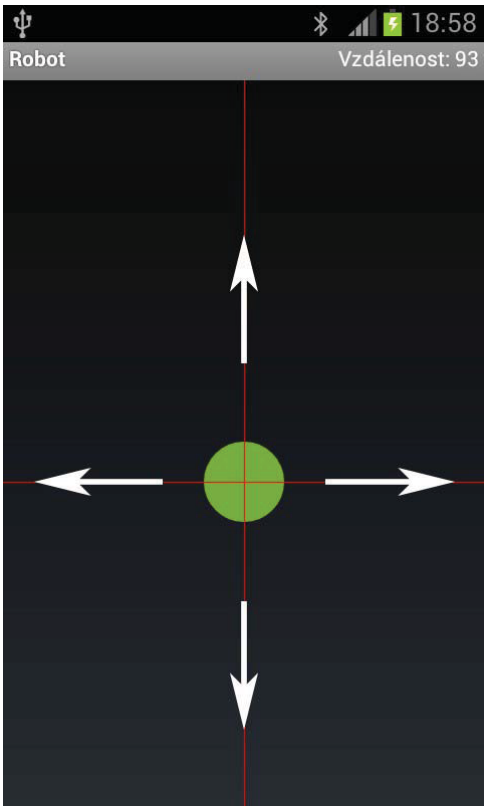

Fig. 4 Mobile application in running mode

C. *Display Resolution*

The application has been designed to adapt to the devices with different screen resolution. For this purpose was used function *getDisplayResolution*, in which is used *getSize* method, which returns the display size in pixels. The display resolution is stored in the format width x height and then used to determine the centre and for display of trackball. Obtained display resolution is also used in class *drive*, where was solving motor control depending on the position of a trackball on the screen of mobile device. This step is important because we want to use for control of the robot always full screen. Because in the application is allowed only portrait orientation, the display parameters cannot be mistaken during the calculation resolution and made an incorrect calculation.

GetSize method can be used for the operating system Android 4.0.3 and higher. Because we had only one device with this version of the system, the application has been only tested for a display with a resolution of 480 x 800 *px*. For devices with different screen resolution will render the graphical part correctly, but class *drive* has not been created, because it would not be possible to test her functionality.

D. *Model Control*

In the Bricx language was necessary to ensure that the engines will be able to work in parallel. Turning the robot was solved by stopping one of the motors which did not allow smooth cornering. This problem was solved by using the synchronization function *Acquire (Release)*.

In the *main* function it was necessary to use the *Precedes* function, to ensure that the tasks are carried out simultaneously, unless there are other addictions. The input parameters of this function are functions to be performed simultaneously. In this case there are functions *drive* and *distance*. In the *distance* function is realized sending of distance of the robot from the barrier.

On the side of the mobile application was created class *drive*, in which is solved controlling of a mobile robot. For the display resolution of 480 x 800 px was driving forward speed limited to maximum *speed* = 130. For movement in the opposite direction is maximum value of *speed* = 357. Making the robot turn was realized by slowing one of the motors. This slowdown is calculated from the variable *Value.* This variable is for straight driving equal to 0. For rotating the model robot left is range of variable *Value* from -1 to -78. For the opposite direction is the range 1 – 78. This variable grows with the increased distance from the vertical axes at the application‘s display. Based on the variables *speed* and *Value* are calculated speed for the left *(LSpeed*) and right (*RSpeed*) motor. These speeds are then sent to the NXT device.

Various positions of trackball with speed that are sent to control unit are shown in (Fig. 4). For rotating the robot on the spot were used the same speed for both engines. According to the direction of rotation is always one of the values reversed. Robot is turns in place, if a variable *speed* takes values in the range of 245-265 and also the values of the variable *value* are not in the range from -10 to 10. The speed of rotation is given by actual value of the variable *Value*.

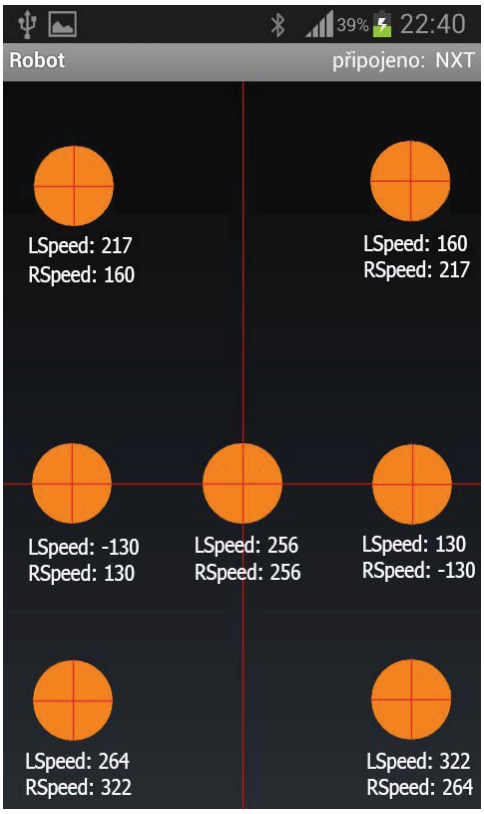

Fig.4 Motor speed for different positions of trackball

E. Ultrasonic Sensor

For distance measurement of the robot from the barrier has been used an ultrasonic sensor which was placed in front of the robot (Fig. 1). Function *SensorUS (IN_1)* in the programming language Bricx is used for reading the data from the sensor. The input parameter of this function is the port to which the sensor is connected. The output value of distance in the range 0-255 is byte type. In case that the distance *l* from the barrier is greater than 255 cm, the sensor sends value 255.

F. Voice Control

Activity for voice recognition is available since the version of the operating system Android 2.2 (*API 8*). The voice recognition for system with a version of Android 4.0.3 (*API 15*) and lower must have access to the Internet because recognition is executed on Google's servers. For *API 16* and higher versions the Internet access is not required and voice recognition takes place offline. The mobile application was optimised for the operating system Android 4.0.3. For voice recognition is the most important function *StartVoiceRecognitionActivity* which works as follows. Class *RecognizerIntent* works with *ACTION_RECOGNIZE_SPEECH* activity that prompts the user for speech and sends input to recognize. Class *RecognizerIntent* sets preferred model for speech recognition and the result is stored in activity *onActivityResult*. In the function *recognized* then method *equals* compares the string *"forward"* with the specified object. If the string is identical with the object, the method returns *true* and the command for move the robot forward is executed. In the case that there is not a consensus method returns *false*, the command is not executed and the application is returned to manual mode.

For controlling the robot by voice was predefined commands *forward*, *backward*, *left and right*, to which the robot responds. Their execution is not time limited and this command can be interrupted with a new command or with touch on the display.

VI. CONCLUSION

The goal of our work was to describe the types of communication between the model and the mobile device and to create a model from Lego Mindstorms that will be controlled using a mobile application. An application for the PC was in the C# language created for testing which was initially programmed for receiving data from the NXT brick. Result of this part was that from the NXT brick was send string of words which has been successfully received in the program created in the PC. Then C# function was programmed to send a text string to the NXT device which is displayed on the screen. By this way was verified bidirectional communication via Bluetooth.

The next step was to connect NXT unit to servomotor and it has been tested for the proper mode. For the best possible driving characteristics we selected mode where accuracy is one degree. From the Lego Mindstorms kit was subsequently built first prototype - model of a car. Using this model there was a problem with the control of the front axle which was

constructed through a system of gears. For this reason there was built another model of robot with a differential gear.

The final step was to program mobile application for the remote control of the robot. The remote control is based on the trackball. The code was written in Java and the display was optimised for a resolution 480x800 px. The model of the mobile robot can be successfully controlled by interaction on the touch display or by voice. When the Android device is connected to Internet, a voice commands can be used in the mobile application to control the robot. Using an ultrasonic sensor, users can monitor the actual distance of the robot from obstacles and display it on the screen of the mobile application.

ACKNOWLEDGEMENT

This research has been elaborated in the framework of the project No: SP 2014/57 - "Autonomous Systems Control and Modern Methods of Diagnostic of the Machines" supported by the Ministry of Education, Youth and Sports.

REFERENCES

[1] GOLDSMITH, Ben. The Smartphone App Economy and App Ecosystems. The Routledge Companion to Mobile Media, 2014.

[2] GOEBEL S, JUBEH R, RAESCH S-L & ZUENDORF A. Using the Android Platform to control Robots, In Proceedings of 2nd International Conference on Robotics in Education (RiE 2011). Vienna, Austria, September, 2011. pp. 135-142. INNOC - Austrian Society for Innovative Computer Sciences.

[3] TIŠNOVSKÝ, Pavel. Eclipse – integrované vývojové prostředí pro Javu i další programovací jazyky. Eclipse – integrované vývojové prostředí pro Javu i další programovací jazyky [online]. 2012 [cit. 2014-01-18]. Available: http://fedora.cz/eclipse-integrovane-vyvojove-prostredi-pro-javu-i-dalsi-programovaci-jazyky/

[4] LEGO Group. NXT User Guide [online]. 2006 [cit. 2013-03-06] Available: http://cache.lego.com/bigdownloads/buildinginstructions/4520736.pdf

[5] JAKEŠ, Tomáš. Řídící jednotka. LEGO MINDSTORMS NXT - Robotické vzdělávání [online]. 2013 [cit. 2014-01-18]. Available: https://www.lego.zcu.cz/web/ridici-jednotka

[6] FRISCHKNECHT, Claudia & OTHER, Thomas. LEGO Mindstorms NXT, Next Generation. [online]. Swiss Federal Institute of Technology Zurich, 2006. [cit. 2014-01-18] Available: http://www.tik.ee.ethz.ch/mindstorms/sa_nxt/index.php?page=tests_us

[7] SMUTNÝ, P. Visual Programming for Smartphones. In Proceedings of 12th International Carpathian Control Conference. Velké Karlovice, Czech Republic, May 25-28, 2011. pp 358-361. ISBN: 978-161284359-9