Control of a Lego Robot with App Inventor for Android

Sergio Sandoval-Reyes and Alfonso Gutierrez-Aldana

Centre for Computing Research. National Polytechnic Institute. Mexico City 07734, Mexico
sersand@cic.ipn.mx; agutierr@cic.ipn.mx

Abstract. App Inventor for Android the blocks programming language of Google and the MIT for developing applications in Android phones and tablets, is becoming very popular among users with no programming experience to wirelessly control robot kits. However the main problem to solve is how to make the Android device to interact with the robot´s motors and sensors. In this paper we discuss three interactive modes (direct, orientation sensor and by speech recognition), to control a Lego Mindstorms NXT Robot using an Android phone.

Keywords: App Inventor, Android, Lego Mindstorms, robot.

1 INTRODUCTION

Android phone and tablets have become ubiquitous devices not just for communication purposes but also to interact with robots due to their computing power and modes of interaction such as touch screen, video and speech. Furthermore, android devices come with numerous integrated sensors such as accelerometer, gyroscope, compass and GPS, and several communication standards like USB, Wi-Fi, Bluetooth, and phone networks (GSM, WCDMA, etc.). In the field of human-robot interaction, the capabilities of a phone to communicate with a robot and specially its sensors (proximity, light, sound, touch, etc.) and actuators (servo motors), become essential.

Although there are many ways to use a smartphone to control wirelessly a robot, most of them requires the knowledge of a classical computer programming language such as C, C++, Java, Python, etc. Lately however, visual programming languages such as Scratch [1], Fujaba [2] and App Inventor for Android [3], can also be used to interact with robots without the need of knowing the classical programming languages.

In this paper, a user interface programmed with App Inventor to control wirelessly a Lego Mindstorms NXT robot using an Android smartphone for human-robot interaction, is implemented. The touch screen and the orientation sensors of the smartphone thus as its speech recognition capability are used to send control commands via Bluetooth to the robot´s servomotors to make it move in several directions.

The remainder of this paper is organized as follows: Section II presents a summary of several approaches to interact with robots. Section III describes the design of the

Advances in Computer Science and Engineering
Research in Computing Science 58, 2012 pp. 253–268
Paper Recived 28-09-2012 and Acepted 07-11-2012

control modes for the Lego NXT robot. Section IV shows the results of the implementation. Our Conclusions are presented in Section V. Finally, Section VI outlines the future work.

2 RELATED WORK

There are many systems to interact with Lego robots which have become very popular in schools and universities, due to their low price and rich assortments of bricks, sensors and actuators to build a great variety of them. For example, the Mindstorms NXT comes with an easy to use graphical programming software to build applications [4]. More recently Lego has offered a free MINDdroid application which is a remote-control application that allows to create a wireless connection directly with the NXT from an android phone, so one can tilt and turn the phone to make the robot move forward, backward and turn to the sides [5]. Naturally the NXT can also be controlled with others alternative firmwares (for example, leJOS for NXT, Scratch for Arduino, Enchanting, etc.), development environments (Scratch, App Inventor) and libraries (Matlab, Python, etc.) for various languages (Java, App Inventor, etc.). Some of the most important are described in the following.

2.1 leJOS NXJ

leJOS NXJ is a Java programming environment for the Lego Mindstorms NXT robots. leJos is an open source project developed by Jose Solorzano that originally implemented a Java Virtual Machine for the Lego Mindstorms RCX system [6]. It consists of: a) Replacement firmware for the NXT that includes a Java Virtual Machine; b) A library of Java classes (classes.jar) that implement the leJOS NXJ Application Programming Interface (API), to write Java programs running on the NXT that interact with sensors and motors; c) A linker for linking user Java classes with classes.jar to form a binary file that can be uploaded and run on the NXT; d) PC tools for flashing the firmware, uploading programs, debugging, and many other functions; and e) A PC API for writing PC programs that communicate with leJOS NXJ programs using Java streams over Bluetooth or USB, or using the Lego Communications Protocol (LCP) to remotely control the NXT. Figure 1 shows a basic movement control ´program and its implementation with leJOS NXJ.

What the program should do:

1. Display "Program 1 " in row 0 of the LCD.
2. Wait for a button to be pressed
3. Run motor A in the forward direction
4. Display FORWARD in the top line.
5. Wait until a button is pressed.
6. Run the motor backward.
7. Display BACKWARD in next line.
8. Wait until a button is pressed.
9. Stop the motor.

```
12 public class MotorTutor1
13 {
14     /**
15      * @param args
16      */
17     public static void main(String[] args)
18     {
19         LCD.drawString("Program 1", 0, 0);
20         Button.waitForAnyPress();
21         LCD.clear();
22         Motor.A.forward();
23         LCD.drawString("FORWARD",0,0);
24         Button.waitForAnyPress();
25         LCD.drawString("BACKWARD",0,0);
26         Motor.A.backward();
27         Button.waitForAnyPress();
28         Motor.A.stop();
```

Fig. 1. A leJOS NXJ program.

2.2 Cellbots for Android

The Cellbots app for Android is the result of the 20%-Google-employee-free-time-project to show some interoperability between Android phones and robot kits [7]. Cellbots was written in Java using the Android SDK and supports four types of robots including the Lego Mindstorms. The Cellbots app available freely in the Play Store Market [8], allows to control a robot to perform forward, backward, turn left, and right movements in four modes: 1) Direct control with D-pad, joystick, accelerometer, or voice control inputs, using a phone-robot Bluetooth connection ; 2) Chat mode using Google Talk, where the phone is mounted on the robot and one chats with the phone-robot

from a laptop, typing simple text messages such as “move forward”, “stop”, etc., through an internet connection; 3) Phone to web mode using a Wi-Fi local network, where the phone-robot will use the phone´s camera to stream video to a browser and from the browser control the robot typing movement commands like “f” for forward, “l” for turn left, etc.; and 4) using a second android phone in which the first one is used as user interface, and the second as a robot controller, using a Wi-Fi local network. Figure 2 shows the cellbots.

Fig. 2. The Cellbots for Android.

2.3 Scratch, Scratch for Arduino and Enchanting.

Scratch is a learning environment developed by the Lifelong Kindergarten Group at the MIT Media Lab, that allows people of any experience, background and age, to experiment with concepts of computer programming, by snapping together visual programming blocks. Figure 3 shows Scratch interacting with sensors of the PicoBoard to make applications that respond to light, sound and touch [10]. The left blocks palette has groups of code fragments (movement, looks, sound, images, control, sensors, motors, operators and variables), that can be dragged onto the scripts area to make programs. Scratch also gave them special form and color. Start blocks

have little knobs (like Lego bricks) under them, to illustrate that there might be placed a block below them. Instructions also have knobs to show how they fit together. Thus, shapes an color of instructions not only give a hint where they can be placed, they even make it impossible to place them at positions that are syntactically incorrect. The third pane of the IDE shows the scene to interact with. On the scene you can create and place new objects. The program controls those objects. The scene also represents the final project. Scratch also includes a "Share" button. Clicking this button publishes the complete project onto the Scratch website and shares it with the community.

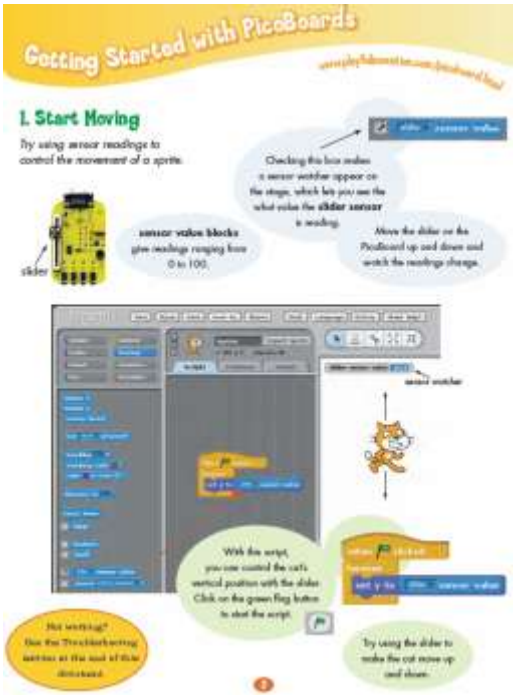

Fig. 3. An application with Scratch and the PicoBoard.

Scratch for Arduino (S4A)

S4A is a Scratch modification that supports simple programming for the Arduino robot. It provides new blocks for managing sensors and actuators connected to the Arduino [11]. Figure 4, shows the Arduino robot.

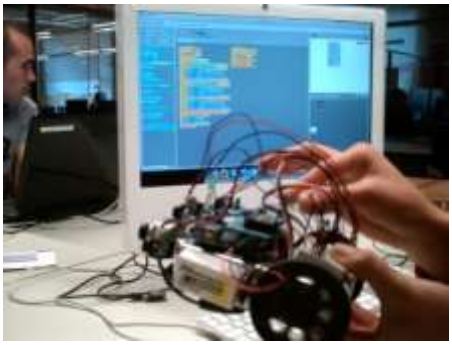

Fig. 4. S4A and the Arduino robot.

Enchanting for Lego Mindstorms Robots.

Enchanting is an easy-to-use-graphical programming language tool for the Lego Mindstorms NXT robots (Figure 4). It is also based on Scratch, and powered by leJOS NXJ [12].

Fig. 5. Enchanting for Lego Mindstorm robots.

2.4 App Inventor for Android (AIA)

AIA is a visual blocks language provided originally by Google and from January 2012, by the MIT Center for Mobile Learning. AIA graphical interface is very similar to the Scratch programming language, allowing users to drag-and-drop visual objects to create applications that run on many mobile phones with the Android OS. [13]. AIA has two main windows: a component designer for building the user interface and a blocks editor for defining the application behavior. Applications can be tested directly on the phone or an emulator (Figure 6).

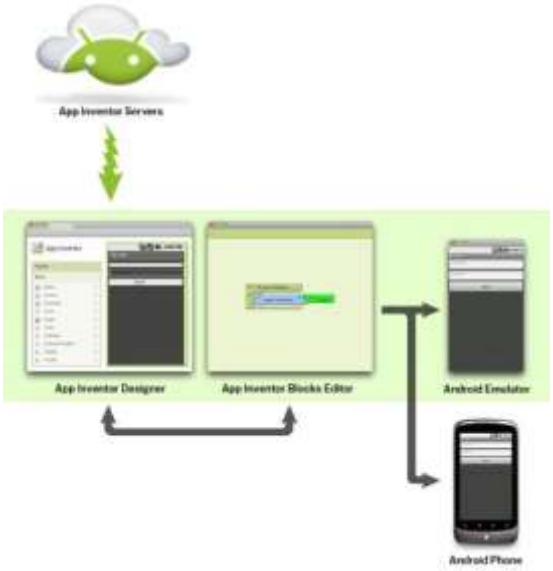

Fig. 6. AIA Component Designer, Blocks Editor & Emulator.-Phone.

The above mentioned solutions (LeJOS, Scratch for Arduino, and Enchanting), to provide additional capability to interact with sensors and motors of a robot, requires to download and replace the original firmware of the robot; a process no very easy because requires certain computing expertise. The Cellbots solution needs also computing skills to compile the source code, unless the downloaded application from the android market is used. Those inconveniences can be avoided using AIA, because AIA includes a Mindstorms palette which provides six Bluetooth client components to interact with sensors and motors of Lego Mindstorms robots. The following section shows three modes to interact with a Lego robot using an Android phone.

3 AIA-Based NXT Robot Control Modes

The app uses touch screen buttons, orientation sensors and the speech recognition capability of an Android phone, for driving the robot forward and backward, turning left and right, and stopping. The app uses also the Bluetooth capabilities of the phone to communicate with the robot.

3.1 Building the User Interface with the Component Designer

The Component Designer shown on Figure 7 is the tool for designing the app interface. The left side palette has two types of components: 1) Visible like button, image, label, ConnectListPicker, etc., and 2) Non-Visible like BluetoothClient, NxtDrive, NxtUltasonicSensor, OrientationSensor, SpeechRecognizer, etc. We drag components from this palette to the viewer to specify the way the phone´s screen will appear when

the application run, for saving data persistently, and for talking to the web. As a component is dragged into an app, its name and type appears in the list of the Components window. Components in this list can be renamed, deleted, and another media added. When a component is selected, its properties that appear in the Properties window, can be modified. The non visible components used in this app are shown in Table 1.

Table 1. Non visible components for the Lego robot control application

Component type	Palette group	Component named as:	Purpose
BluettothClient	Other stuff	BluetoothClient1	Connect robot
NxtDrive	LEGO MINDSTORMS	NxtDrive1	Drive the robot´s wheels
NxtUltrasonicSensor	LEGO MINDSTORMS	NxtUltrasonicSensor1	Detect obstacles
Notifier	Other stuff	Notifier1	Display error messages
OrientationSensor	Sensors	OrientationSensor1	Detect orientation of phone
SpeechRecognizer	Otherstuff	SpeechRecognizer1	To control the robot by voice

Fig. 7. AIA Component Designer for the Lego robot app.

The visible components to create the user interface are shown in Table 2. These components are also shown in the components window of Figure 7. *ListPicker* displays a list of the robots that have been paired with the phone and lets you choose one. The upper and lower buttons connect and disconnect the robot via Bluetooth. The five middle buttons are for direct control. The blue left button is for orientation sensor control, and the blue right button is for voice control.

Table 2. Visible components for the Lego robot control application

Component type	Palette group	Component named as:	Purpose
ListPicker	Basic	ConnectListPicker	Choose the robot to connect to
HorizontalArrangement	Screen Arrangement	HorizontalArrangement2	A visual container
Image	Basic	Image1	Robot and phone image
VerticalArrangement	Screen Arrangement	VerticalArrangement1	A visual container
HorizontalArrangement	Screen Arrangement	HorizontalArrangement1	A visual container
Button	Basic	OrientationSensorButton	
Button	Basic	DirectControlButton	
VerticalArrangement	Screen Arrangement	VerticalArrangement2	A visual container
Button	Basic	ForwardPitch	Drive forward
Button	Basic	LeftRoll	Turn left
Button	Basic	RightRoll	Turn right
Button	Basic	BackwardPitch	Drive backward
Button	Basic	ForwardButton	Drive forward
HorizontalArrangement	Screen Arrangement	HorizontalArrangement	A visual container
Button	Basic	LeftButton	Turn left
Button	Basic	StopButton	Stop
Button	Basic	RightButton	Turn right
Button	Basic	BackwardButton	Drive backward
Button	Basic	SpeechRecognizerButton	To activate the speech recognizer
Button	Basic	DisconnectButton	Disconnectrobot

3.2 Programming the Behavior with the Blocks Editor

The behavior of the application is defined in the Blocks Editor. This editor is launched clicking the "Open The Blocks Editor" button in the Component Designer. This Editor has two palettes from which blocks are dragged, the Built-in palette and the My Blocks palette (Figure 8).

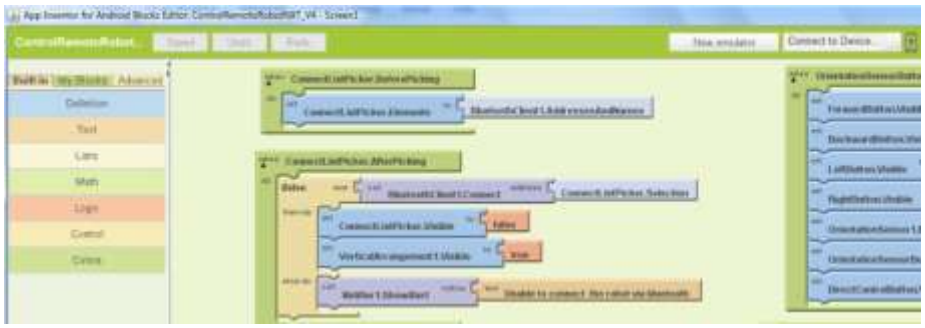

Fig. 8. The Blocks Editor with the Build-in and My Blocks palettes.

The Built-in palette contains built-in blocks for standard programming control and functionality, for text and list manipulation, and mathematical, logical and control operators (Figure 8). The My Blocks palette contains blocks representing the components of the applications that were added in the Component Designer. In a very similar way like Scratch, the App Inventor blocks language provide visual cues to ease the development tasks, and only some blocks lock in place, reducing the possibility of errors (Figure 8). The application behavior is directly defined through a set of event-handlers (e.g., "when event ForwardButton.Click occurs, do NxtDrive.MoveForward"). Live testing can be performed with a plug in phone or with an Android phone emulator, clicking the *Connect to Device* button or the *New Emulator* button located in the upper right side. Once the application is tested, it can be deployed by packing it into an Android app by clicking the upper right *Package for Phone* button in the Component Designer (Figure 7).

3.3 Connecting and disconnecting the Lego Robot

The first behavior is connecting to the Lego robot. This is done dragging the *ConnectListPicker.BeforePicking*, and the *ConnectListPicker.AfterPicking* components from My Blocks palette (Figure 9). When in the phone the *Connect* the Robot via Bluettoth *button is clicked*, *the* BeforePicking *component will show in the phone a list of paired robots detected, and let you to choose one. When you choose a robot, the* AfterPicking *component, will make a Bluetooth connection to that one*. *The* Disconnect Robot *button on the phone triggers the* DisconnectButton *component to disconnect it (Figure 9).*

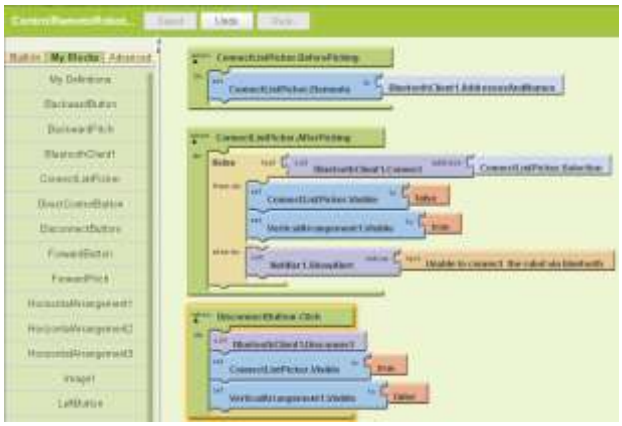

Fig. 9. Connecting and disconnecting the Lego robot.

3.4 Driving the Lego robot

As it was mentioned, the Lego robot can be controlled in three modes: Direct, Orientation Sensor and by Voice. The design of each mode is described in the following sections.

A. Direct Control

This is the default mode with the five middle buttons (labeled ^ for forward, *V* for backward, < for left, > for right and *stop*), shown in Figure 7. The NxtDrive component provides the five blocks for driving the robot´s motors (Figure 10) at 90% power [14].

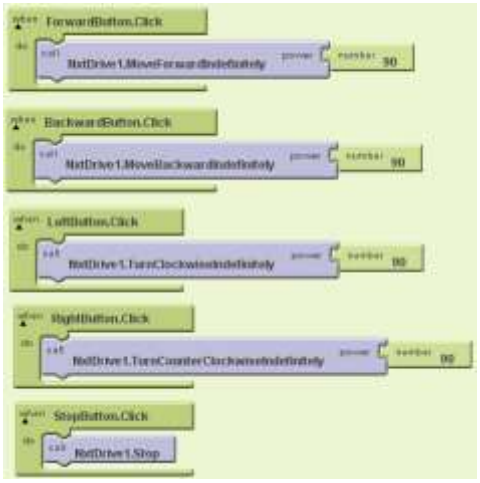

Fig. 10. Driving the Lego robot with Direct Control.

B. Orientation Sensor Control

The non-visible *OrientationSensor* component (Figure 7) is used for game-like apps in which the user controls the action by tilting the phone [14]. It can also be used to control the robot´s movements clicking the blue left button on the phone. The *OrientationSensor* has five properties two of which are used for this app. These are: *Pitch* and *Roll*.

The *Pitch* property is used to perform forward-backward movements. Having the phone in horizontal position (Pitch of 0 degrees), a forward movement is performed tilting the phone so its top is pointing down (Pitch increases to +90 degrees). A backward movement is performed tilting it so its bottom points down (Pitch decreases to -90 degrees).

The *Roll* property is used to perform left-right movements. A turn-left is performed when the phone is tilted up onto its left side (Roll increases to +90 degrees). And a turn-right when it´s tilted up onto its right side (-90 degrees).

The implementation for move forward and turn-right is shown in Fig. 11. Move backward and turn-left are similar.

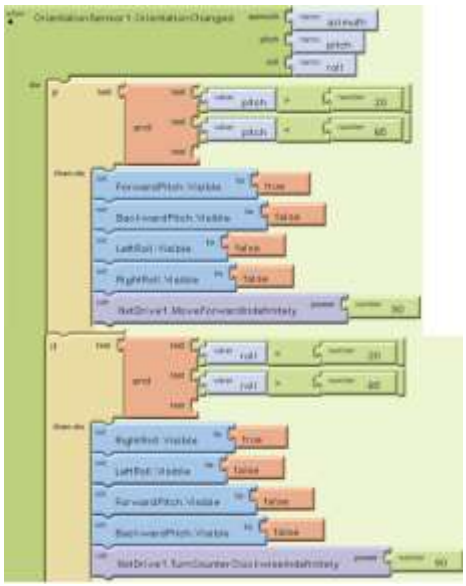

Fig. 11. Orientation sensor control mode.

C. Voice Control

This mode uses the *SpeechRecognizer* component of AIA and is triggered clicking the blue right button on the phone. The *SpeechRecognizer* uses Google´s network-dependent voice-to-text system to transcribe user´s vocal input like “move forward”, “turn left”, “stop”, etc [15]. The component requires the phone´s internet connectivity. Figure 12 shows the *VoiceControlButton* calling the speech components and labels being populated with possible text results like *forward*, *backward*, *left*, etc.

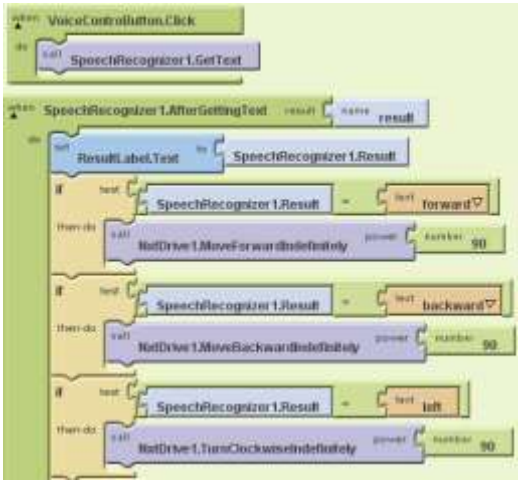

Fig. 12. Speech recognition control mode.

The *.GetText* call initiates the AIA speech component, which prompts the user for input and then sends the sound clip to Google´s speech-to-text system. The resulting text is sent back to the phone to trigger the *.AfterGettingText* event to drive the robot´s motors accordingly.

4 Experiments and Results

For the experiments we use an old Lego Mindstorms robot and a LG-P500 Android 2.2 phone, with the *Voice Search-Android* Google application [16]. Figure 13 shows the results of implementing the proposed AIA-based system to interact with the robot in three control modes. The left figure shows the interface of the default control mode with buttons to connect and disconnect the robot via Bluetooth. The middle figure shows the interface in orientation sensor control mode. The figure toward the right shows the interface in speech recognition control mode.

The three control modes performed well. The direct control mode was the best, followed by the orientation sensor mode and the voice control mode. In the first mode the Lego robot responded rapidly to each movement command sent from the Android phone. The orientation sensor mode was a little bit more difficult because it was necessary to control more precisely the pitch and roll tilting of the phone. A slight tilting of the hand holding the phone produced undesired movements. The third mode, the voice control had the worst performance due to two problems. The speech recognizer component no always recognized appropriately a voice command, sometimes instead of recognizing the “forward” command, it recognized it as “for all”. But, and this was the second problem, the resulting text of the recognized command had a response delay of three to five seconds, which was the time it takes to send the sound clip to the Google´s speech-to-text system, and getting the resulting text back.

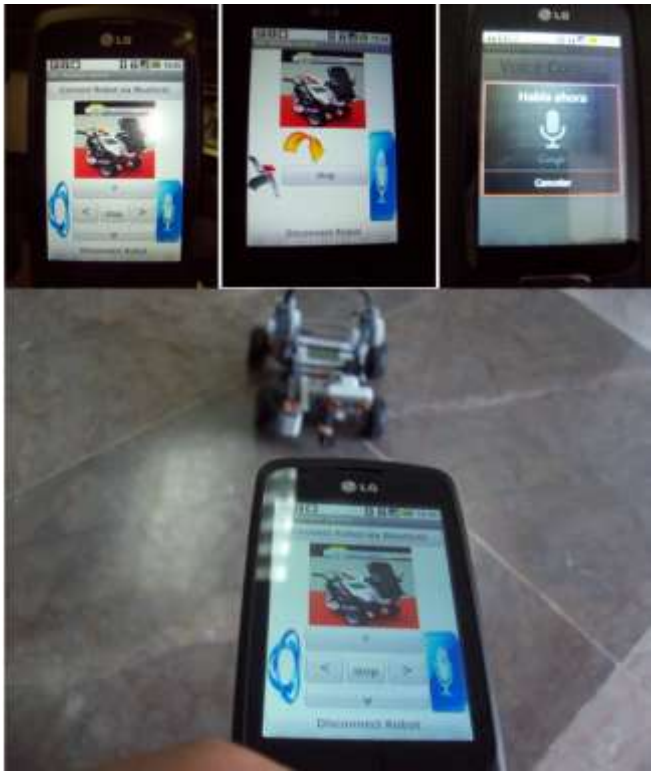

Fig. 13. Control modes of interacting with the Lego robot.

5 Conclusions

The Android OS for smart phones is gaining a greater share of the phone market. App Inventor for Android targets a wide audience who want to create mobile applications in their phones, but that no necessarily is knowledgeable of a computer language like Java. App Inventor eases the development of phone applications using a graphical interface with intuitive drag-and-drop features that is very similar to MIT´s Scratch. In this paper an App Inventor based application to interact with a Lego robot in three control modes was implemented. It showed that the robot´s actuators can be easily controlled by touch-screen buttons, the sensor orientation and the speech recognition of the Android phone.

6 Future Work

This application may still be expanded beyond the actual state, making it more useful by attaching the phone to the robot and using its camera to stream video back to a web browser, for internet mode control.

Acknowledgements

This work was supported by SIP-IPN project 20121288.

References

1. Scratch imagine program share, *Scratch: a programming language for every one*. http://scratch.mit.edu , 2012.
2. R. Jubeh, "Simple robotics with Fujaba," in *Fujaba Days*. Technische Universitat Dresden, Sep. 2008.
3. Wikipedia, *Google App Inventor*, http://en.wikipedia.org/wiki/Google_App_Inventor, 2012.
4. Lego Mindstorms Robots, http://mindstorms.lego.com .
5. Lego Mindstorms, *Do Androids Dream of Lego Mindstorms?*. http://mindstorms.lego.com/en-us/News/ReadMore/Default.aspx?id=227417 , 2012.
6. leJOS, *leJOS NXJ: Java for Lego Mindstorms*. http://lejos.sourceforge.net/nxt/nxj/tutorial/Preliminaries/Intro.htm , 2012.
7. The official Google Code Blog, *Android streches its legs… err wheels… with help from 20% time at Google*, http://googledevelopers.blogspot.mx/2010/122/android-stretches-its-legs-err-wheels.html .
8. Cellbots, *Cellbots:Using Cellphones as Robotic Control Platforms*. http://www.cellbots.com , 2012.
9. Google play, *Android Applications: Cellbots*. https://play.google.com/store/search?q=cellbots&c=apps 2012.
10. Scratch Store, *Picoboard and Lego We Do Robotics Kit*. http://info.scratch.mit.edu/Scratch_Store , 2012.
11. Projecte Scrastch, *Scratch for Arduino*. http://seaside.citilab.edu/scratch , 2012.
12. Enchanting, *Graphical Programming Tool for Lego Mindstorms NXT Robots, based in Scratch*. http://enchanting.robotclub.ab.ca/tiki-index.php , 2012.
13. MIT Center for Mobile Learning, *App Inventor for Android*. http://www.appinventor.mit.edu , 2012.
14. David Wolber, Hal Abelson, Ellen Spertus and Liz Looney, **App Inventor, Create Your Own Android Apps**, Ed. O´Really, pp 185 & 324, 2011.
15. Jason Tyler, **App Inventor for Android, Build Your Own Apps No Experience Required!**, Ed. John Wiley, 2011.
16. Google play, *Voice Search-Android On Google Play*. https://play.store.com/apps/details?id=com.google.android.voicesearch , 2012.