Android-Based Low-Cost Robot Controller
Christoph Krofitsch and Reinhard Grabler
Practical Robotics Institute Austria (PRIA)
krofitsch@pria.at, grabler@pria.at

Android-Based Low-Cost Robot Controller

1 Introduction

Since humanity discovered fire, we simplify life by using technology. Today new achievements in the field of technology seem to be faster than ever before. There are so many situations in life, where technology is essential - at school, at work or just because of convenience. To simplify our lives even further, complex applications are invented which require high computing capacity. Therefore it is important to produce high-performance hardware which is smaller, faster, and environment-friendly as well. Robotics in education is seen as an interdisciplinary, project-based learning activity drawing mostly on math, science, and technology and offering major new benefits in education at all levels [1, 3]. Robotics implements 21st century technologies and can foster problem-solving skills, communication skills, teamwork skills, independence, imagination, and creativity [2, 3]. Just like the smartphone market over the past few years [4], robotics could boom if it would be more appealing for personal usage. Like computers, smartphones became parts of our lives in the meantime because they are powerful, personalized and easy to use. There are many companies on the free market, which encourage each other to remain being innovative to sell their products, causing new technologies evolve rapidly. To cause the trend go to personal robotics (analog to personal computing), the society has to be convinced by providing a robotics set that is powerful, affordable and easy to use. The Botball Educational Robotics Program uses the CBCv2 robot controller (referred as CBC), based on a Chumby. This controller is well suitable for the Botball program and STEM education with no doubt.

In the Botball competition and educational robotics, the CBC's power is largely enough, as the success of this program proved. But in other robotics applications which require more power, it may not be sufficient. The project "Disbotics" of the Vienna University of Technology used the CBC as a basis for a rule-based multi-agent framework. Initially, their robots had very long reaction time which was insufficient for their use case [5]. Botball provides a game kit that also includes cameras that can be connected to the CBC and used with the built-in computer vision system. Unfortunately, the vision system on the CBC is difficult to use because of slow frame rates and high light sensibility leading to color recognition issues.

The idea of this project is to take advantage of the smartphone market trend and make common smartphones to usable robot controllers, benefiting from their technical capabilities and familiarity to the society.

2 Our Approach

Our goal is to develop an Android-based robot controller which is

- **powerful:** Through the high processor performance in most Android phones, even robotics applications with require high computing power may be realized.

- **easy to use:** Android is an widely spreaded and stable operating system, providing open interfaces for peripherals. With help of a microcontroller board, we ensure compatibility to Botball sensors and actuators.
- **affordable:** By using a device that most people already own, the costs can be significantly reduced.

Using the camera and other built-in sensors make it to a good basis for robotics. Consequently, every Android user having a Botball kit and using our project can start doing robotics. Our approach consists of the Android phone, an Arduino microcontroller board with a specially designed shield, and a development machine, as presented in figure 1.

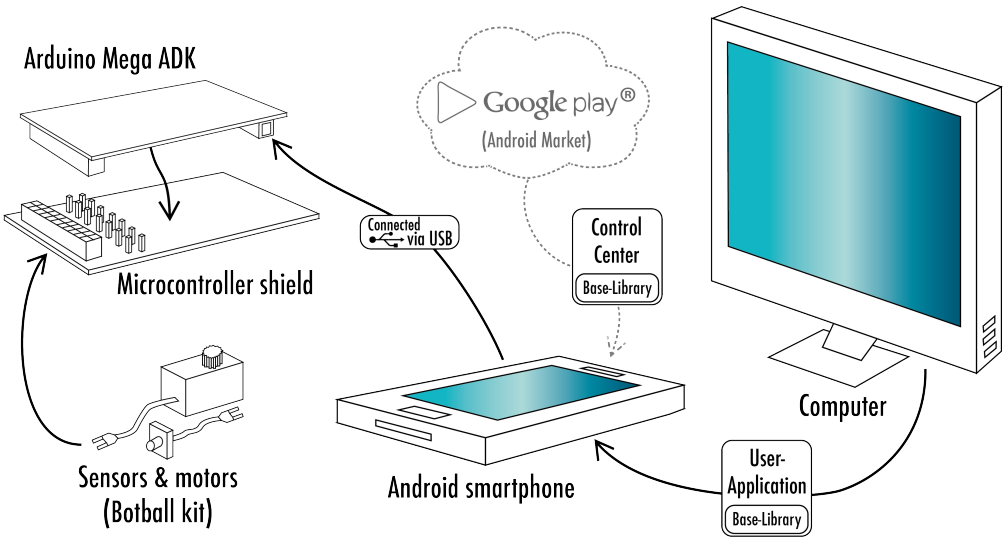

Figure 1: The approach - a system overview

The several components of this approach are explained detailled in the subsequent chapters.

2.1 Arduino Prototyping Platform

Considering that Botball sensors and actuators can't be connected to a common Android phone directly, we need an approprate microcontroller board for realizing this project. This board must have a sufficient number of ports for connecting sensors/actuators as well as an interface to Android phones to somehow make the sensors and actuators accessible for the phone. The Arduino Mega ADK board [6] is well suited for this matter. Arduino is an open source physical computing platform consisting of a set of microcontroller boards and simple development environment for writing programs for them [7].

2.2 The Shield

A shield is a device put on an microcontroller board for adding functionality to it. In our case, we need a shield in form of an electronics board for mapping the input/output ports of the Arduino board appropriately for Botball sensors and actuators, because the ports of the Arduino Mega ADK don't match them. The shield is intended to provide as much comfort and functionality as the CBC does. The ports of the Arduino board [6] can be mapped to 10 analog, 16 digital, 6 motor and 6 servo ports. For ensuring an appropriate voltage supply and taking off the load of the phone battery, an additional battery pack will be installed.

2.3 Arduino Software

At the first glance, it looks like that every robotics task would require a different software for the Arduino board because of different I/O handling. Nevertheless, this would result in higher effort for the user as he has to write code for the Arduino board for sensor/actuator handling as well as for the Android phone. In order to avoid that, our approach is to write a multifunctional and flexible software for the Arduino board which basically remains on the board. This software represents a middleware between sensors/actuators and the Android phone and must be able to (1) read sensor values and forward them to the phone, (2) power actuators according to commands from the phone and (3) handle this as efficient as possible. This can be reached by standardized communication interfaces between the Arduino board and the Android phone. In other words, the software on the Arduino board cares about the sensors and actuators, enabling the user to keep focused on programming the phone.

2.4 Android Software

The software concept for the Android phone includes the communication with the Arduino board, a helper app and the actual software for the robot. It is described subsequently.

2.4.1 Base-Library

The Base-Library handles the communication to the Arduino board and is responsible for getting desired sensor values, powering the right actuators and, in essence, providing the user a comfortable library for making robotics applications. For the communication to Arduino, it uses the Android Debug Bridge (ADB), because of its broad support for different Android versions [8]. The software is realized as a library, which means that it is not a standalone Android app but can be used in other apps. It is the base of the following two software concepts.

2.4.2 Control Center

The user interface on the CBC provides control about the connected components, which makes it comfortable for the user to read sensor values, test actuators and calibrate the camera. To provide exactly this on the Android phone as well, we develop a similar software which we call Control Center. It is a downloadable Android app that comes with the Base-Library. Primary goal of this app is it to provide easy configuration interfaces to support the development of the User-Apps. In other words, it is not mandatory for programming the robot, but a good assistance.

2.4.3 User-Application

The user-written applications (User-Apps) contain the logic for the specific tasks, similar to programs developed in the KISS-IDE [9]. The process of programming should be as easy as possible, reachable by considering two factors: (1) the programming language and (2) the IDE. Due to the Android software architecture, the programming language must be Java which is considered as very beginner-friendly. Because the Eclipse IDE as the common IDE for Android apps requires a bit know-how and may seem complicated to beginners, a much simpler IDE is planned to be used: the Processing IDE [10]. With this IDE, the user can focus on the program logic without worrying about Android software components and their behavior. The Base-Library will be included in the IDE and uploaded to the Android phone automatically as part of the program. As a result, the user can directly start up writing the program, upload it to the phone, plug in the Arduino board and execute his code as an app.

3 Comparison of Usability: CBC vs. Android Controller

In this section we will compare the usage of the CBC robot controller and the Android controller. It shows what the user actually has to do for programming his robot. The following figure 2 shows a schema of the program development process in both variants:

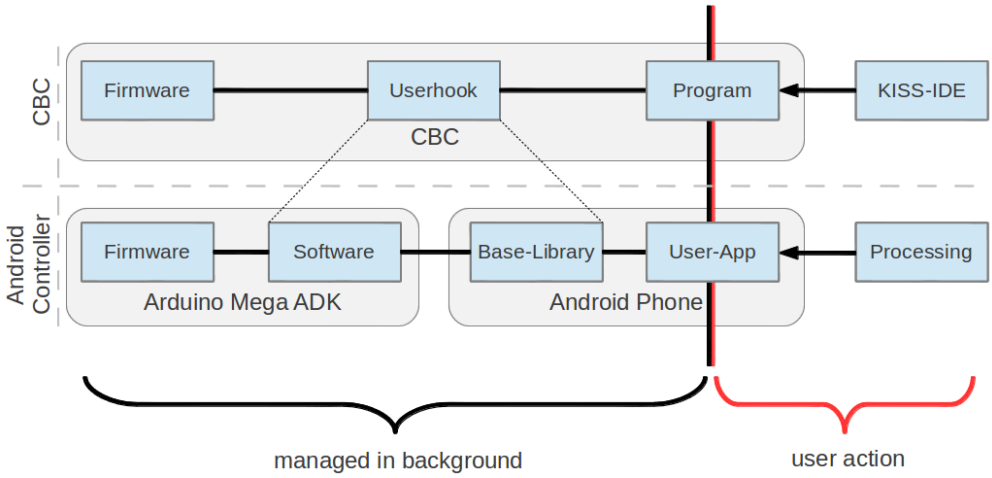

Figure 2: Schema of the program development process for the CBC and Android controller

For the case the user is using the CBC, he develops the program in KISS-IDE, loads it onto the CBC and executes it from the GUI. The userhook - a software running on the CBC's firmware - shows the GUI and provides the interfaces to sensors and actuators.

For the case the user is using the Android controller, he writes his code in the Processing IDE. By pressing one button, the program gets built to an Android app and automatically uploaded to the phone. In order to start his code, he has to connect his phone to the Arduino board via USB and start the app. In the background the Base-Library is running, automatically managing the sensors and motors by communicating with the Arduino board.

As a result, the effort for the user writing a program for his robot is more-or-less the same for both controllers: actually it's a button for deployment and another one for starting the code.

4 Implementation

4.1 The Interplay of Arduino and Android Software

One of the most difficult parts in the implementation is keeping the communication between the Arduino board and Android phone as efficient as possible. This means, that the sensor values and motor position feedback should only be sent when really required. We solved that with the following approach: When the User-App calls a method for a sensor or motor value, a request command will be sent to the Arduino board and it responses immediately with the desired value. This approach is effective and requires two communication ways for getting one value, which is applicable for this matter. The communication for powering actuators is simple: when invoked in the User-App, a command will be sent to the Arduino board which immediately powers the actuator. Stopping and other actuator actions work similar.

4.2 Java Object Model

Everyone who is familiar to the CBCJVM [11], a framework for programming the CBC in Java, can start off programming his User-App. The available Java classes and interfaces for programming the robot are similar to those of the CBCJVM and work almost equally. Additionally, some classes will be added for the built-in devices of the Android phone for accessing them directly. For showing its simplicity, the following code drives a motor at port 0 until the digital sensor at port 10 is pressed.

```
Motor m = new Motor(0);
Digital d = new Digital(10);
m.moveAtVelocity(1000);
while(!d.getValue());
m.off();
```

Listing 1: Example: drive a motor until digital sensor is pressed

4.3 Control Center

As already mentioned, the Control Center helps the user developing his application. In the implementation we try to stick to the GUI of the CBC as it is simple and most Botball students are already familiar with it. The following diagram (figure 3) shows the main navigation points of the control center:

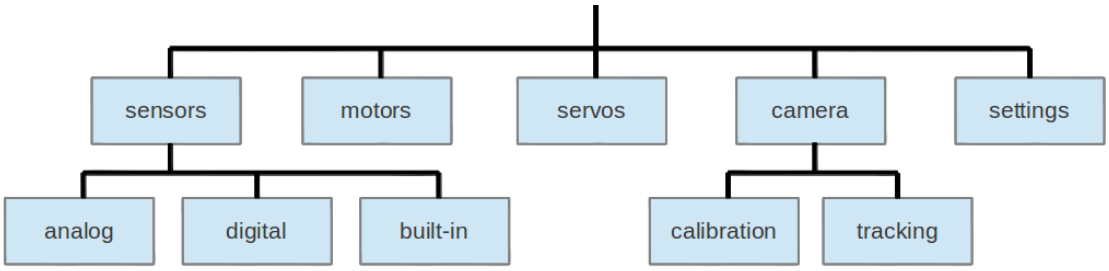

Figure 3: Control Center navigation

The built-in sensors we support in the first version are the accelerometer, gyroscope, light, magnetic field, linear acceleration and proximity. In future versions we plan to integrate more features of unique Android devices (e.g. 3D object recognition for devices with a 3D camera) and add some useful development tools such as events, predefined values, etc.

5 Conclusion

The variety of robotics is significant - it ranges from industrial robots, over autonomous vacuum cleaner, up to humanoid robots. This field of technology may look very complicated and incomprehensible, but this does not have to be this way. This project aims to make people think different about robotics. It makes it possible to develop robots easily and control them with a device nearly everyone already owns: a smartphone. The kit consists of an Arduino board and a special shield for connecting sensors and actuators from the Botball kit directly. Additionally, a simple IDE and a tutorial is provided to start off writing software for the robot. What the user has left to do is to connect his phone to the board, execute the app on the phone, and watch his robot in action.

For future work, we plan to improve our project. These improvements are including support for the iRobot Create, a wireless connection between the phone and the Arduino board (bluetooth), possible integration of user-written programs into the Control Center instead of standalone applications, and the adaption to iOS to reach more potential users. Furthermore, we would be very glad to cooperate with KIPR and make the KISS-IDE to the primary IDE for this project.

6 Acknowledgement

At first, we want to thank our project leader and mentor Dipl.-Ing.(FH) Mag. Gottfried Koppensteiner who made all of this possible. Special thanks to Werner Schnöll for his great support in developing the shield and the Austrian Federal Government for their program "Sparkling Science" and their financial support. Additionally, we want to thank Dr. David Miller and Steve Goodgame for their responsiveness and good will. Also, thanks to Matt Roman for making the wiring diagram of the CBC available.

References

[1] Alimisis D., Karatrantou A., Tachos N.: "Technical school students design and develop robotic gear-based constructions for the transmission of motion", Digital Tools for Lifelong Learning, Proceeding, Warsaw: DrukSfera, Eurologo 2005, pp. 76-86

[2] Karatrantou A., Tachos N., Alimisis D.: "Introduction in basic principles and programming structures using the robotic constructions LEGO Mindstorms", Proceedings of the 3rd national Conference, Teaching Informatics, University of Peloponnese.

[3] Sergeyev A., Alaraje N.: "Promoting Robotics Education: Curriculum and State-of-the-Art". the Technology Interface Journal/Spring 2010.

[4] J. Angelo Racoma: CMS Wire. `http://www.cmswire.com/cms/web-engagement/google-ramps-up-mobile-advertising-efforts-launches-mobile-now-initiative-010173.php` Retrieved 2012-05-29.

[5] Mongia H., Esberger M.: "DISBOTICS". Submitted to the GCER12.

[6] Arduino: Arduino Mega ADK Board Page. `http://arduino.cc/en/Main/ArduinoBoardADK` Retrieved 2012-05-28.

[7] Arduino: Introduction. `http://arduino.cc/hu/Guide/Introduction` Retrieved 2012-06-04

[8] Android Developers Guide: Android Debug Bridge. `http://developer.android.com/guide/developing/tools/adb.html` Retrieved 2012-05-30.

[9] KISS Institute for Practical Robotics: KISS-IDE. `http://kipr.org/products/kisside` Retrieved 2012-06-04.

[10] Processing Hompage: `http://processing.org/` and Processing for Android Wiki Page: `http://wiki.processing.org/w/Android`. Retrieved 2012-06-04.

[11] McDorman B., Woodruff B., Joshi A., Frias J.: "CBCJVM: Applications of the Java Virtual Machine with Robotics". Presented at GCER 2010.