Cheap, Easy Robotics for the Non-Programmer

James K. Larson[1], Adeilton C. Oliveira Jr[1], Victor M. Oliveira[2], Brett Nelson[3], Josue J. G. Ramos[4], Lucas T. Alves[2]

[1]Babuino Project,
[2]Br-gogo Project,
[3]Faubion Gradeschool, Portland, Oregon, USA
[4]Renato Archer IT Center, Brazil
jlarson@pacifier.com, adeiltonjunior@terra.com.br, victormatheus@gmail.com, brettn@teleport.com, josue.ramos@cti.gov.br, ltanure@gmail.com

Abstract. This paper presents an alternative for cheap and easy robotics that integrates an Arduino based hardware, Babuino, with a graphical programming environment known as Blocos, and a very cheap mobile robot, Babuinobot, giving the non-programmer the opportunity for building robotics experiments using a graphics Logo dialectic, and allowing him to become digitally fluent.

Keywords: Software, hardware, robotics, environment, pedagogical.

1 Introduction

The objective of this paper is to present an inexpensive (nearly free) programming environment that allows people to control a robot without the programmer realizing he is actually programming. This environment is composed of a graphical programming software and an Arduino based hardware.

The first question is: "How is this system different from Arduino [1] or Lego Mindstorms [2] since they claim to make Robotics easy?"

What's wrong with Legos? The main problem with Lego Mindstorms is that it costs from $200 to $400, which puts it out of reach for a large number of people. One of the real strengths of the original Mindstorms was the programming environment. Since it was designed with kids in mind, it provided an intuitive, graphical method of programming. Recently in the NXT version it has become bloated and is somewhat intimidating.

What's wrong with Arduino? The Arduino is very attractive since its hardware is cheap and the software is free, so it is affordable for people to download and install the software, and make an led or two blink, with a simple example program. For a person that is not a geek, the C-like syntax and use of libraries can be overwhelming, and trying to modify somebody else's library code is simply too complex for the beginner.

Proceedings of SIMPAR 2010 Workshops
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Darmstadt (Germany) November 15-16, 2010
ISBN 978-3-00-032863-3
pp. 465-473

The solution: suppose there is a way to combine the inexpensive Arduino hardware with an intuitive programming environment like Mindstorms? There is! And the software is public domain, open source (it is free) , Fig. 1 shows this system. For teachers and educators, there's even free coursewares. It doesn't get much better than that.

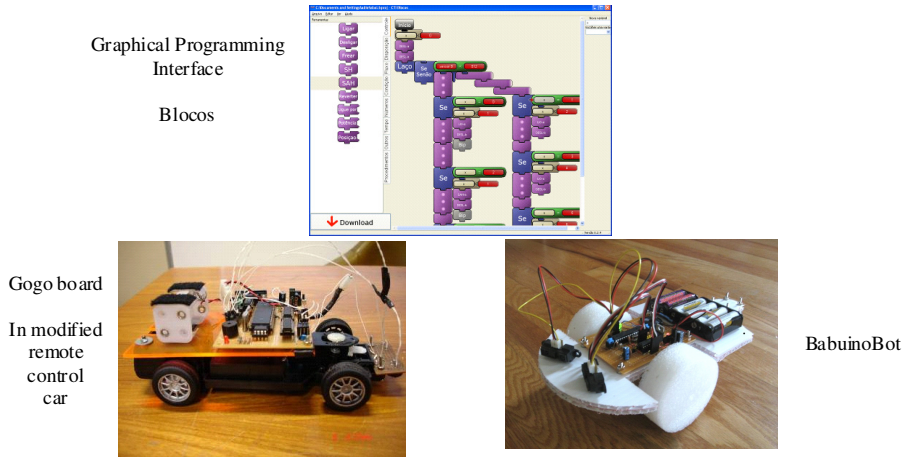


Fig. 1. Gogo board and Babuino can be programmed by Blocos

The Graphical Programming Environment is known as “Blocos” and the robot hardware is known as “Babuino”. This paper will discuss them, how to get them, and how to have fun with them. Babuino / Blocos is in use in Brazil, Spain, Argentina, Colombia, and the US.

This paper presents how to turn an Arduino into a programmable robot controller and how to set up a graphical programming environment for it.

2 Logo Programming

Here's an abbreviated and imperfect history of fun-to-program robotics. Back in the late 60s, Seymour Papert [3] at MIT decided that robots should be fun. His excuse was that he wanted to help children learn. One of his concepts was known as Turtle Graphics in which an imaginary turtle could be programmed to draw lines on the screen. Shortly afterward, the idea of making a motorized turtle that could drag a pen on a piece of paper evolved. MIT got Legos involved and the first Programmable Brick (robot controller) was created. Mindstorms (name taken from a Papert book) followed quickly. A spin off, based on the work of Fred Martin, was a programmable brick known as Cricket [4]. Later a free system called Gogo board became available. Most recently, an inexpensive brick, known as Babuino and based on Arduino, has been developed.

To go along with his easy to use hardware, Papert invented a language called Logo [5] to make programming equally easy. Logo was supposed to be easy for anyone

Proceedings of SIMPAR 2010 Workshops
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Darmstadt (Germany) November 15-16, 2010
ISBN 978-3-00-032863-3
pp. 465-473

(adults included) to learn. It was easy compared to languages of the day, but still involved a text editor and memorizing syntax. Papert's collaborators created a graphical environment called Logo Blocks [6] in which blocks representing Logo statements could be metaphorically snapped together to build the logical constructs of the robotics programs. Fortunately, Logo Blocks is much easier to understand than that last sentence describing it! Logo is now available as Cricket Logo and is the basis for Logo Blocks. Cricket Logo and Logo Blocks are both only for Windows and Mac environments. Blocos is a modern replacement running on Linux, Mac and Windows. One additional concept is integral to the Programmable Brick and the Logo Blocks environment. To keep the Programmable Brick simple and cheap, the software that runs on it must be very simple. So the brick software knows nothing about the elegant programming environment provided by Logo Blocks. Instead it accepts single byte codes, also known as "opcodes". Each opcode translates to exactly one simple action that the brick software will perform. For example, opcode 46 might refer to Motor A while opcode 49 might turn on a motor. So the opcode sequence 46 49 would cause Motor A to turn on. The result is that Logo Blocks (and Cricket Logo also) knows nothing about the Programmable Brick; it simply generates opcodes. The brick knows nothing about Logo Blocks; it simply responds to the opcodes sent to it.

A direct consequence of this clever scheme is that new languages and environments (such as Blocos) may be developed as long as they produce opcodes as their output. An equally important consequence is that new programmable bricks (such as Babuino) may be created as long as they provide appropriate actions in response to a stream of opcodes.

3 Babuino

On the hardware side, besides Lego Mindstorms, simpler and cheaper versions of the Programmable Brick, known as Crickets, are available. But even these are not exactly cheap. So a still cheaper brick, known as the GoGo Board [7], intended for aspiring roboticists in the developing world, was created by MIT and a team in Brazil working together. They also created Blocos [8], the graphical programming environment presented in this paper. The only drawback to GoGo is that you have to etch the circuit board and round up all the parts yourself, and etching a circuit board is beyond the skill level of most children and beginner roboticists.

The newest alternative is Babuino [9]. Babuino uses the very inexpensive and popular hardware platform known as Arduino (or a Freeduino [10] clone) which can be purchased cheaply with no assembly required. You can save more if you can solder and build your own Arduino from a kit, but it's not required. And if you get tired of Babuino (as if that will ever happen!) and want to do other things, you still have an Arduino! As you might quickly surmise, Babuino is just software for an Arduino. When you download it to your Arduino, it turns the Arduino into a Programmable Brick capable of interpreting opcodes. Certain pins are defined as inputs and outputs for the brick; add some simple hardware to interface motors and sensors, and now you have a very inexpensive robotics experiment system!

Proceedings of SIMPAR 2010 Workshops
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Darmstadt (Germany) November 15-16, 2010
ISBN 978-3-00-032863-3
pp. 465-473

Babuino is the Programmable Brick half of your robotics experiment system. The other half is the software environment. We use Blocos Brazilian-developed, block-oriented programming environment that makes robots simple to program. Blocos is really easy to use. Fig. 2 shows 4th grade students programming Babuinos with Blocos.

Fig. 2. 4th grade Students at Faubion Pk-8 school, Portland, Oregon, USA

3.1 Creating your own Babuino

To create your own Babuino you only need two things: An Ard/Free-duino and the Babuino software. Obviously, if you already have an Arduino, you can load Babuino software onto it and you are ready as in the left of Fig 3. Have a look at this Wiki [10] article to learn more about Ard/Freeduino and locate some suppliers. The simplest version is suitable to become a Babuino. You can even build your Freeduino on a breadboard [11] as in the right of Fig 3. In [12] and [13] are two alternative methods of building an Arduino.

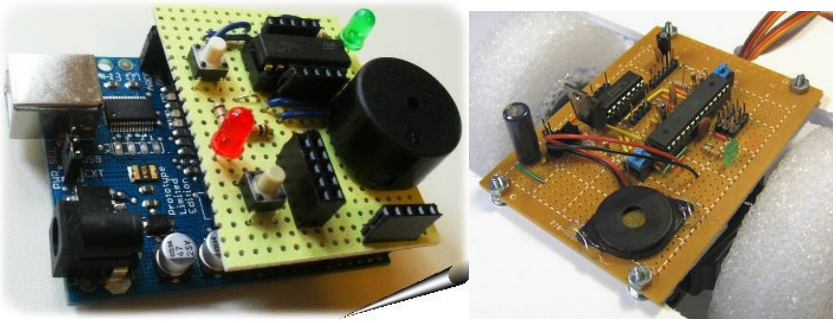


Fig. 3. Left arduino + expansion to Babuino, right an entirely hand made Babuino

Download Babuino from the Support Forum [9]. You need to sign up (free, easy; and if you have a Yahoo login, you're half way there), then you can access the Files Page. The figure of the Files Page shows the files you need to download. Get the Command Reference which is a complete manual for using Blocos (which we'll

Proceedings of SIMPAR 2010 Workshops
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Darmstadt (Germany) November 15-16, 2010
ISBN 978-3-00-032863-3
pp. 465-473

download in the next section) on Babuino. Click on Firmware to go to the page with the code – the Firmware Page is also shown here. Right click on the latest .hex file and save it in an appropriate location. This is what you'll upload to your Arduino.

To actually upload Babuino to your Arduino, follow these directions [9]. Just be sure to specify the hex file you uploaded with the latest firmware, not version v01.hex. Babuino is not actually Arduino code, but can be loaded using the Arduino bootloader and does not harm the Arduino bootloader. (So when you're done with Babuino, you still have an Arduino.) To do the upload, you'll need a USB to serial adapter. A lot of stores offer a nice, reasonably priced one if you don't already have one. You'll be able to use it to download your Blocos programs to Babuino as well. If you already have your Arduino working with a serial port, it'll work just fine for Blocos/Babuino.

You should know that Babuino is still being developed, so while there'll be more features in the future, everything works that you need to make a robot go. There is active bug fixing and expansion. You can join the discussion and provide your inputs on the support forum that you just signed up for.

Babuino defines certain pins of the Arduino to do certain functions. These include controls for two DC motors, four switch or sensor inputs, a beeper, a serial port (the Arduino serial port) to talk to a pc (Linux, Mac, or Windows), a RUN/STOP button (guess what it does), a User Led and a Status Led. Just the resources needed for a small robot! The connection diagram is shown. You need to add the RUN button, the Status Led, and the beeper to your Arduino to make it into a Babuino.

4 The Programming Environment: Blocos

Blocos is one of dialects of Logo, with graphical capabilities. The main reason for this development is the unavailability of a free and open platform for programming robotics kits [14]. It is based on Logoblocks with the main purpose of helping inexperience users to develop the concepts involved in programming and robotics [8]. Scratch [23] is a very nice system offering a block programming environment that can also be used to easy learning of programming. Scratch has interface to boards, as Gogo Board, to turn on lights or motors, which can be used to make a kind of robot. The main issue is that it only runs in tethered mode, so that it always requires a cable connecting the computer to the robot, and its autonomy is reduced.

Fig. 4. Left: IF statement in Portuguese. Right: Procedure in Blocos

Proceedings of SIMPAR 2010 Workshops
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Darmstadt (Germany) November 15-16, 2010
ISBN 978-3-00-032863-3
pp. 465-473

In Blocos every component of the Logo Language is mapped in a block, and one block can be only connected to a compatible block, that is, the “IF” block can only connect with the block of Boolean expression and so on. Fig. 4 shows these properties. In the left and the possibility of writing graphically procedures in the right

Blocos is written in Python [19], so installing it is a two step process. First, download and install Python, then download the Blocos Python Program itself. It's an easy process and you can do it in either order. There's a nice discussion forum for Blocos [15], just like for Babuino. Sometimes the same issues are discussed, but the two are separate and distinct. You'll probably want to sign up for both of them. Since these two projects are based in Brazil, some of the discussion is in Portuguese or Spanish - but there's lots in English as well. If you post in English, you'll be answered in English.

You'll find the download page here [16]. For all systems, download the Blocos Python program indicated. It's a tar file so you can use either 7-zip (Windows) or gzip and tar (Mac or Linux) to unpack the files. (There's also a zip file for Windows – either one works.) Create a directory for it. Move the zip file into that directory and either unzip it or untar it. Windows only, download the Python Package that is indicated. You install it in the usual manner; you have to be Administrator to do the install. As the install proceeds, you'll have to respond to several prompts. Take the defaults, agree to licenses and mostly just say yes. You may get an error or two about files not found, but you should just ignore these. Blocos works on XP, W2K, Windows vista and W7. Try it and post problems to the Forum.

To install Python on Linux, you will use apt-get in a terminal window. You will need to be super user or use sudo. Enter the following command at the prompt:
“apt-get update”.
Then enter this one at the next prompt:
“apt-get install python2.5 python-cairo python-gtk2 python-kiwi python-serial”.
Relax for a minute while the systems grinds away and you're done installing Python.

For Installing the Python package on Mac is best done using MacPorts [17]. Install MacPorts if you haven't already done so. MacPorts allows you to safely install software on your Mac without worry about interfering with any system software. Directions for using MacPorts are here [18]. Following the directions, install the following in the order given: python-cairo, python-gtk2, python-kiwi, and python-serial. (I haven't done this myself. Others who have tell me this works fine.) You'll also need a soft link to your USB serial port. That'll look something like: sudo ln -s /dev/tty.complicatedName /dev/ttyUSB0 ... where "complicatedName" means the (unfortunately unique) name assigned to your FTDI cable.

Once the Python Package is installed and Blocos is unzipped (or untarred), simply double click (on Windows) the Blocos.py file. In Linux (hopefully Mac also) from a terminal window, cd to the directory containing Blocos.py. Start the program by entering “python Blocos.py” (without the quotes). The programming window which you'll see is shown in the figure. The figure is annotated to show the key components of the interface.

Proceedings of SIMPAR 2010 Workshops
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Darmstadt (Germany) November 15-16, 2010
ISBN 978-3-00-032863-3
pp. 465-473

5 The BabuinoBot

You can build an inexpensive robot platform following the instructions on this web site [20] which shows how to build an Arduino clone using a kludge board. This version includes all the hardware interface to have a Babuino.

The Babuinobot is a robot that uses the Arduino microcontroller to run the Babuino software which allows the robot to be programmed using CTI Blocos. Costs vary considerably depending on bulk purchase discounts, availability of local materials, re-using parts and materials from discarded waste and electronics, etc. One was able to construct 6 robots using all new materials for about $35 US each, the instruction are in [21]. The basic material for building the robot are corrugated plastic that is often used for signs because it is lightweight, plus modified RC servomotors, double sided foam tape, a 1¼ "caster wheel that is widely available in the US, some screws and that is all . Fig. 5 shows steps in building BabuinoBot

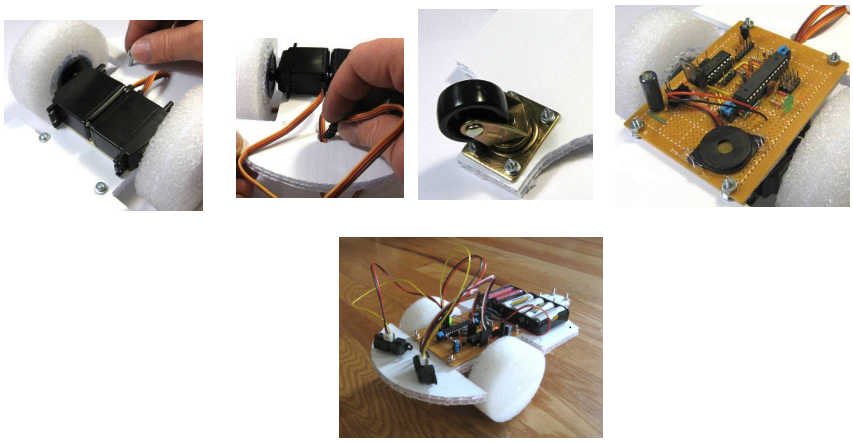
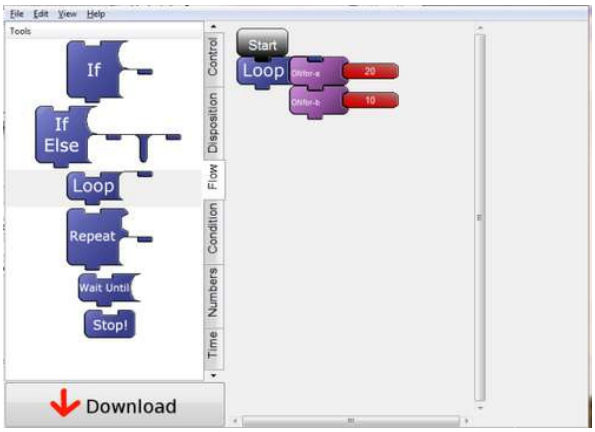


Fig. 5. Steps for building a BabuinoBot

6 Creating a BabuinBot Program Using Blocos

Fig. 6 shows a simple program in Blocos. Although the blocks seems to be self explained, the meaning is described in the sequel. After the program is downloaded to your Babuino and the RUN button is pressed, Motor A is turned on for 2 seconds (20 tenths of a second), then it stops and Motor B is turned on for 1 second. Being in a loop, this behavior continues until the RUN button is pressed again or the batteries run dead.

Proceedings of SIMPAR 2010 Workshops
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Darmstadt (Germany) November 15-16, 2010
ISBN 978-3-00-032863-3
pp. 465-473




```
to start
    loop [
            a, onfor 20
            b, onfor 10
    ]
end
```

Fig. 6. Shows in the Left the blocos interface and the equivalent Logo Program generated by Blocos

Create the program just described. Simply click on a component in the Blocks Pane, move your cursor into the Program Pane, and the component will appear under your cursor. Click again to drop it, and use the Define Pane to set the parameters. Click the tabs to select different sets of program blocks. The blocks fit together to create the program. If you have any problems, read the Command Reference you already downloaded.

Once you have created the program, the next step is to download it to your Babuino. You can use the same USB to serial interface cable you used to upload Babuino itself. From the Blocos program, simply click the large Download button. Blocos will search for serial ports and will test to see if a Babuino is connected. When Blocos finds a suitable port, it downloads your program. Babuino will beep when the download is complete and Blocos will show a suitable message.

As the program is downloaded, it is stored in the EEPROM memory of the Babuino. So once the download is complete, you can disconnect the USB cable (not required, but your robot won't go far with the cable connected). You have to have a power supply (probably batteries) connected. Press the RUN button on the Babuino and your program should run. Press the RUN button again to stop the program. You can even turn the power off and back on, hit the RESET, then hit RUN again. Your program should run. You'll note that the Status Led on Babuino will blink at different rates when it is running (fast blink) and when it is stopped (slow blink).

To load another program, simply reconnect the USB cable and use Blocos to download the new code. Be sure you stop any running program (Status Led should be doing a slow blink).

6 Conclusion

The purpose of this paper was to explain how to set up a really cheap robotics development system that's fun and easy to program. If you've followed the steps, you've turned your Arduino into a Babuino and are now using Blocos to program it.

Proceedings of SIMPAR 2010 Workshops
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Darmstadt (Germany) November 15-16, 2010
ISBN 978-3-00-032863-3
pp. 465-473

We have provided some links to give you ideas on how to build your own Arduino-based robots that can be programmed with Blocos. Besides the photo galleries I've already mentioned, you can see a Babuino/Blocos robot in action here [22]. Your robotics experiments are now limited only by your imagination, not by your hardware and software skills.

Acknowledgments. The development of Blocos was sponsored by CNPq, that sponsored the students of the Scientific Initiation Program of IT Center Renato Archer that developed Br-Gogo and Blocos.

References

1. Arduino Home Page , http://www.arduino.cc; http://en.wikipedia.org/wiki/Freeduino; 2010,
2 Lego; Lego Home page, http://www.lego.com; 2010.
3 http://en.wikipedia.org/wiki/Seymour_Papert
4. Cricket (2009) "Handy Cricket Home Page". Available at: <http://www.handyboard.com>. 2010.
5. http://en.wikipedia.org/wiki/Logo_(programming_language)
6. Begel, A. (1996) "A Graphical Programming Language for Interacting with the World". < http://research.microsoft.com/~abegel/mit/begel-aup.pdf>. 2010.
7. Sipitakiat, A.; Blikstein, P.; Cavallo, D. (2004) "GoGo Board: Augmenting Programmable Bricks for Economically Challenged Audiences", In: Proceedings of the Int. Conf. of the Learning Sciences (ICLS 2004), Los Angeles, USA.
8. Ramos, J. J. G.; Silva, F. A.; V. Oliveira; Alves, L. T.; D'Abreu, J. V. V. "Development of open hardware and software components for low cost pedagogical robotics programs". In: Anais do IX Simpósio Brasileiro de Automação Inteligente (SBAI 2009), Brasília, DF. In Portuguese
9. http://babuinoproject.blogspot.com/
10. http://en.wikipedia.org/wiki/Freeduino
11. http://dl.dropbox.com/u/4930890/Construction/English/Babuinobot_Circuit_Board_1.0.pdf
12. http://www.instructables.com/id/Perfboard-Hackduino-Arduino-compatible-circuit/
13. http://itp.nyu.edu/physcomp/Tutorials/ArduinoBreadboard#toc9
14. Ramos, J. J. G.; Azevedo, H.; Vilhete V. A. J.; Neves O.; Figueiredo, D.; Tanure L.; Holanda F. (2007) "Initiative For An Open And Low Cost Pedagogical Robotics For The Social And Digital Inclusion In Brazil". In: Anais do VIII Simpósio Brasileiro de Automação Inteligente (SBAI 2007), Florianópolis, SC. (in Portuguese)
15. http://groups.google.com/group/br-gogo?pli=1
16. http://sourceforge.net/projects/br-gogo/files/
17. http://www.macports.org/
18. http://guide.macports.org/
19. http:// www.python.org/
20. http://home.teleport.com/~babuinobot_a/index.html
21. dl.dropbox.com/u/4930890/Construction/English/Babuinobot_Assembly_Instructions.pdf
22. http://blip.tv/file/3699384
23. http://scratch.mit.edu/