Enhancing the Polulu 3pi with RobotBASIC

John Blankenship
Samuel Mishal

ISBN 9 781438 276052

The source code for the program listings in this book (and much more) is available to the readers at:

http://www.RobotBASIC.com

Contents at a Glance

Table Of Contents

Preface

RobotBASIC, a free Robot Control Language, has an integrated robot simulator with numerous sensors, making it easy to develop complex robot algorithms and behaviors. This book explains how to modify and expand the standard Pololu 3pi robot so that it has many of the capabilities of RobotBASIC's simulated robot. It also explores several example projects utilizing the modified robot in order to demonstrate how to exploit the new features.

If you have not programmed in RobotBASIC before, please visit our web page

www.RobotBASIC.com

to download your free copy of RobotBASIC as well as a short PDF Tutorial on using the language. RobotBASIC comes with nearly 300 pages of documentation, but if you are totally new to programming you might consider one of our introductory texts such as *RobotBASIC Projects for Beginners* or *Robots in the Classroom*, both of which are discussed and available on our web page. If you need more detailed information on developing algorithms for robotic behaviors, consider our advanced book, *Robot Programmer's Bonanza*.

The project described in this book is not suitable for beginners. The construction of the robot described here, assumes you are familiar with soldering techniques and can read and interpret schematics. If you do not possess these skills you can easily DESTROY the 3pi robot. It would be

nice if Pololu could offer either a kit or fully assembled version of the robot described in this book and we will certainly help them pursue such an endeavor should they feel the demand warrants it. If you would be willing to purchase such a product, please let them know at **www.Pololu.com**.

Finally, we realize this is a daunting project that is not suitable for many non-technical hobbyists. For that reason, we are constantly working on ways to make it easier for everyone to use RobotBASIC with real-world robots. Visit our web page for the latest news and innovations.

Chapter 1

Overview

Although the RobotBASIC simulated robot provides a very powerful and rewarding programming experience, for many hobbyists, a real physical robot is the ultimate goal. Unfortunately, most robot kits provide only a few types of sensors at best, and if you've programmed the RobotBASIC simulator, you know you cannot give your robot intelligent behaviors without an appropriate number and variety of sensors. If you have not programmed with RobotBASIC before, it is suggested that you read the PDF Tutorial and other information available at **www.RobotBASIC.com**.

The sensors (types, number of, and locations of) provided on the RobotBASIC simulator were not chosen lightly. They provide everything necessary to develop a wide variety of robotic behaviors. It follows then, that an ideal hobby robot would have the same sensors as the RobotBASIC simulation and offer transparent compatibility with the RobotBASIC language.

Building such a robot can be a daunting task, even with today's technology. Initially, we considered building a robot totally from scratch, but when we found the Pololu

3pi, we knew we had a nearly perfect platform for a powerful educational or hobbyist-oriented robot.

The standard Pololu 3pi (see Figure 1.1) comes assembled, with five line sensors and the ability to read the battery voltage. Even though that does not sound like a lot, the line sensors allow the implementation of algorithms to follow lines and negotiate line mazes. For more information on the 3pi, visit **http://www.pololu.com.**

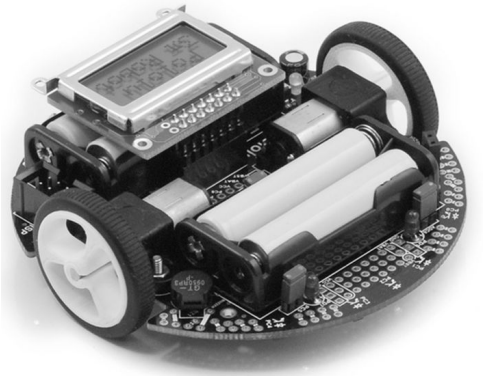

Figure 1.1: The Pololu 3pi is a small, yet ideal, starting platform for creating our robot.

Notice that the 3pi is round, just like the RobotBASIC simulation. Pololu provides software that offers a *slave* mode that makes it easy to control the 3pi from an external computer or microcontroller. While such a mode would make some implementation tasks easy (and can certainly be accomplished using the RobotBASIC serial commands), it does not provide the power nor the flexibility truly required for this project.

1.1 RobotBASIC's Communication Protocol

RobotBASIC provides a unique built-in serial protocol for wireless adapters (such as Bluetooth) that allows programs developed with the simulator to immediately control a real robot. Let's see how this works.

Normally, when you want to tell the simulated robot to move, you need a command like `rForward 20` which tells the simulated robot to move forward on the screen twenty pixels (negative numbers move backward). A command such as `rTurn 15` tells the robot to turn 15º to the right (negative numbers turn left).

If you want these same commands to control an external real robot, you simply have to issue the command `rCommPort N` (at the beginning of your program). This command indicates that a Bluetooth adapter (or other communication device) has been connected and is using serial port **N** (typically a Virtual Serial Port). Just replace **N** with the actual number of the port being used - more on this later.

Once the `rCommPort` command has been issued, all robot commands (including `rForward` and `rTurn`) no longer control the simulation. Instead, they automatically send two bytes to the specified serial port. The first of these bytes is an operation code that identifies the command. The table in Figure 1.2 shows the code used for each of the simulator commands we will be implementing.

Command	Op-code
rLocate	3
rForward	6
(backward)	7
rTurn (right)	12
(left)	13
rCompass	24
rBeacon	96
rRange (right)	192
(left)	193
rSpeed	36
rChargeLevel	108

Figure 1.2: These are the RobotBASIC commands (and their op-codes) that will be implemented on the 3pi.

The second byte sent to the robot is either zero (if not needed) or a parameter related to the command. For example, when the command `rForward 20` is issued, the PC will send out a 6 followed by a 20. The 6, of course, indicates the FORWARD operation is being requested and the 20 specifies how far forward to move.

In addition to the above commands, RobotBASIC provides an `rCommand` function that allows the RobotBASIC programmer to implement custom commands. In a later chapter, we will see how this command can provide a number of unique capabilities.

In addition to receiving commands from the PC, the external robot has to return sensory data to RobotBASIC. The robot returns five bytes of sensory data *every* time it receives a command. Three pieces of data (information from the bumpers, the infrared object detection sensors, and the line sensors) are very time sensitive and are nearly always returned in the first three of these five bytes (in the order listed above).

The remaining two bytes are usually zero because they are typically not needed. When the commands `rCompass`, `rBeacon`, and `rRange` are executed though, these two bytes are used to return the requested data.

When RobotBASIC receives these five bytes, it automatically extracts the individual pieces of information and uses them appropriately. For example, the `rBumper()` function normally provides the status of the simulated robot's bumpers. After an `rCommPort` command has been issued though, `rBumper` will return the status of the real robot's bumpers because RobotBASIC will automatically use the last data received from the external robot.

1.2 Capabilities to be Implemented

When fully implemented, this provides a system where the external robot has nearly all of the capabilities of the simulation. These capabilities are summarized below.

- 4 Bumper sensors (front, rear, left, and right)
- 5 IR perimeter proximity sensors (all equally spaced on the front half of the robot)
- 5 IR Line sensors (beneath the front edge of robot)
- 1 Electronic compass (accurate to 1 degree)
- 1 Beacon detector (capable of recognizing 15 different beacons)
- 1 Battery-level sensor
- 1 IR distance-measuring sensor (1-30 inches)
- 1 Servo controlled turret for the distance-measuring sensor
- Full motor control through `rForward` and `rTurn`

1.3 The Remote Robot's Responsibilities

It is important to realize that although RobotBASIC automatically sends out the appropriate data and properly utilizes the information returned, it is the robot's responsibility to actually carry out the requested actions and supply the necessary sensory data.

All of the robot's responsibilities will be handled by a control program written in C. The Pololu web page provides information on how to install a free C compiler on your PC and how to download the compiled files to the 3pi. Chapter 3 will examine the robot's new control program in detail. **It is important to realize that once this program has been written and installed on the robot, it never again has to be modified or downloaded because the robot itself will effectively become an extension of RobotBASIC**.

Before we can explore the robot's program though, we must construct the actual robot hardware, which will be examined in Chapter 2.

1.4 Summary

In this chapter, you have learned:

- About the Pololu 3pi robot and why it was chosen as the base platform for this project.
- How RobotBASIC's built-in wireless protocol communicates with external robots while maintaining compatibility with the integrated simulator.
- Which of RobotBASIC's simulated sensors will be implemented in the physical robot.