1
CHAPTER

Assembly Instructions

When I was first approached by McGraw-Hill to develop a simple training robot, I came up with the following description:

- Differential drive robot, 5" long by 4" wide
- Robot electronics built onto a standard 0.060" epoxy/fiberglass printed circuit board (PCB) that is used as the robot's chassis
- Powered by a single 9-volt alkaline or rechargeable radio battery
  - 20+ minutes powered operation per battery
- Dual H-bridge for both motors running at 20-kHz pulse-width-modulated motor controller frequency
  - 4 Pulse-width-modulated power levels along with full stop
  - Full speed approximately same as walking
- Collision detection using two wire "Whiskers" mounted on the robot
- Dual CDS cells for light level measurement
- Robot peripherals (motors/IO) controlled by Microchip PIC16C505 microcontroller along with an implementation of simple robot "behaviors":
  - Default random movement behavior (with collision avoidance)
  - Selectable "photovore" behavior with stop on collision
  - Selectable "photophobe" behavior with stop on collision
  - Selectable wall-hugging/maze-solving behavior
  - Pulse-width-modulated speed select (off + 4 levels)
- BS2 socket with "AppMod" socket and BS2 programming connector
  - Two-wire interface between robot's PICmicro MCU and control BS2. This interface is designed to use standard BS2 "SHIFTIN" and "SHIFTOUT" commands.
- Target customer: 14+/high school student

These specifications were a result of what *I* thought was the ideal robot. For the most part, the basic concept has remained true to form with the final *TAB Electronics Build Your Own Robot Kit*.

If you compare these proposed features with the design of the robot that came with this kit, you'll see that virtually all of these requirements have remained. The

only two significant changes to these specifications are the change from two wire "whiskers" to two infrared proximity sensors and the addition of an infrared remote control that simplifies the use of this robot. The end result is quite an impressive package that I'm sure you will have many hours of fun and learning with.

Even if you do not match the profile of the target customer, chances are you are new to robotics, and many of the robot specifications listed above are unfamiliar to you. Before discussing how the robot is assembled, I would like to explain the different features of the robot and what they mean in relation to the *TAB Electronics Build Your Own Robot Kit*. I will expand on all of these points in later chapters, but for now I would like to introduce you to different features of the robot.

In Figure 1-1, I have labeled a photograph of the robot with each of the major components identified. As I work through the different features of the robot, I suggest that you come back to this photograph to see where the components are located on the robot PCB.

There are several different ways of making a robot move. Chances are you have seen a TV show or movie in which a robot "clanks" around on two feet somewhat like a human. While there are some robots that are built from the "humanoid" model, the *TAB Electronics Build Your Own Robot Kit* is moved by simple electric motors turning wheels. There are two wheels (as is shown in Figure 1-2) that can turn together (to move the robot forward or back) or in opposite directions to turn the robot.

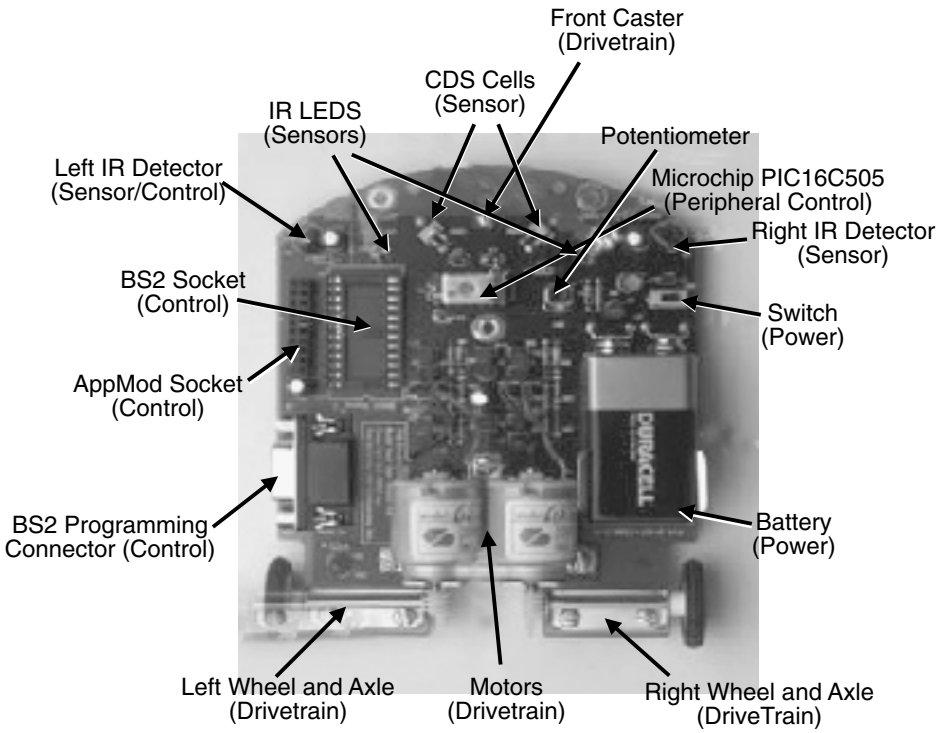

Figure 1-1 Robot layout with parts identified.

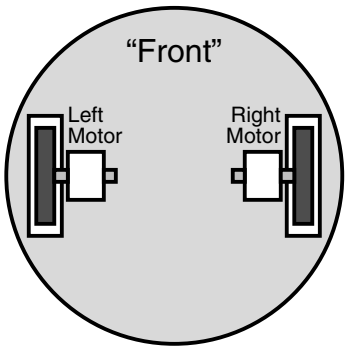

Figure 1-2 Differentially driven robot.

Many robot kits, including "toy" robots, are built with plastic chassis that mounts electronics, motors, and drivetrain components. This approach was not used for the *TAB Electronics Build Your Own Robot Kit* for two reasons. The first was the cost that the development of molds would have added to the product. A secondary concern was the tolerances that have to be built into plastic moldings. These would have probably resulted in "tweaks" to the design to make sure that the parts fit together under all foreseeable circumstances.

Instead of using molded plastic, the decision was made to use the fiberglass PCB material for the robot chassis. This material is very tough, and it holds its dimensions very well. In addition, metal stampings that could be bolted to the fiberglass PCB were used to hold the motors and drivetrain components (gears, axles, and wheels). The result is a robot that may look somewhat less "polished" than most "toys," but one that is far more robust, will last longer, and is less likely to have problems during use than an all-plastic product.

An important consideration for this robot was to keep the power options simple. In response to this, a single 9-volt "premium" alkaline or NiMH rechargeable battery should be used with it. As I will discuss in later chapters, the choice of battery can be very important: It can cause the robot to be unable to run at all (using a carbon battery), and it can make it run the best (fastest and longest) using either a premium alkaline battery or a rechargeable nickel-metal hydride.

The best batteries I have found as I write this (August 2001) are the Eveready Energize e^2 and the Duracell Ultra III. For rechargeable batteries, I have found that the Radio Shack NiMH batteries (Catalog Number 23-299) work very well. Please note that battery technology is continually changing, and there may be some new products and technology released since I wrote this—don't be afraid to try different products, especially if they are new.

The 20-minute "life" of the robot may seem quite short, but you will find that it is adequate if you are testing out a new application. When the motors are stopped, the major drain on the battery is the PICmicro microcontroller (or the BASIC Stamp 2, if it is installed). As you work with your robot, you will find that you might want to have two or three rechargeable batteries so that you can work a reasonable amount of time each day.

The "H-bridge" motor controller allows the motors driving the wheels to be turned on and off and move in forward and reverse directions independently. By

rapidly turning the power to the motors on and off, the speed at which the robot runs can be controlled. The *TAB Electronics Build Your Own Robot Kit* has five speed levels—from completely off to full "on." In the later chapters I will explain in more detail how the motor drivers work and how they are controlled.

When the robot was first designed, we wanted to use pieces of wire (known as "whiskers" or "feelers") to detect objects in front of the robot as well as act as "bumpers" when the robot collided with an object in front of it or to the side. We had a *lot* of problems with the wire whiskers—it was difficult to make them sensitive enough for the application and robust enough to not be damaged during normal operation and handling. Plus, they were very expensive.

The solution to this problem was to look at different technologies for collision sensors. Recalling the playing around I did when I got my first TV with an infrared remote control (like bouncing the signal off a wall to the TV), I tried to implement a simple system that used this capability for the *TAB Electronics Build Your Own Robot Kit*. To my surprise, this method of detecting objects worked very reliably, and despite the need to add a second IR detector, the cost of this solution was quite a bit less than the cost of the wire whiskers.

The IR collision sensors work by sending a short, low power "burst" of infrared light out of an IR light emitting diode (LED) close to the IR detector as shown in Figure 1-3. The power output of the LED is controlled by a potentiometer on the robot itself. The distance at which objects can be detected can vary from a fraction of an inch to many inches.

Calling the IR LED and detector combination a "collision sensor" is really an inaccurate description, as they will be set to detect objects several inches away to give the robot an opportunity to stop without colliding with the object. I prefer the term "proximity sensor" since it detects objects within a certain range or proximity.

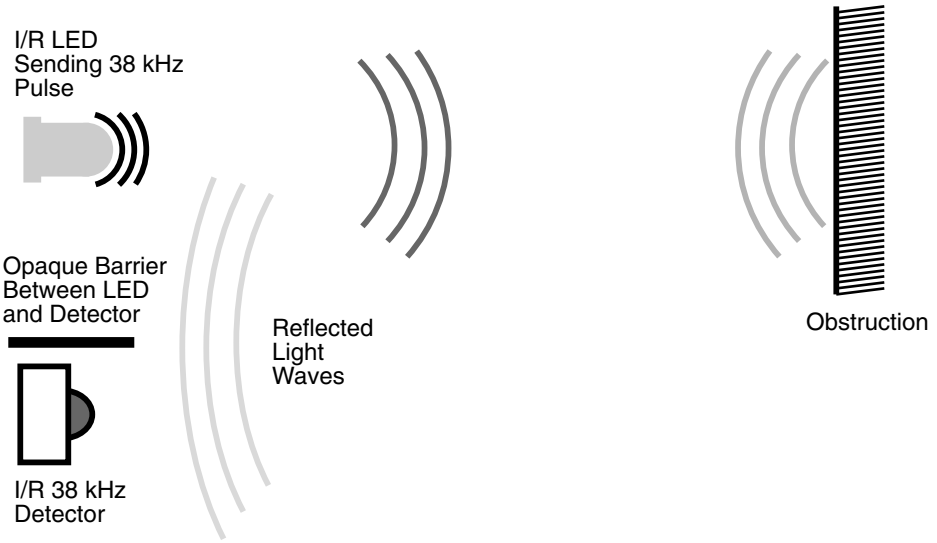

Figure 1-3 Infrared object detection.

The proximity sensors give the robot a very basic ability to "feel" objects around them. This sense is complemented by the two CDS cell "light level" detectors that are designed to allow the robot to determine which direction has the brightest light. The detectors are required for the "Photovore" and "Photophobe" behaviors, and the light levels they are exposed to can be read by a BS2 installed in the robot.

I have written three books about the Microchip PICmicro® microcontroller families, so it should not be a big surprise that the primary controller built into the robot is a member of this family. The PIC16C505 built into the robot performs what I will refer to as "peripheral" functions, controlling the ability of the robot to move, integrating the sensor (proximity sensors and CDS cell light level detectors) information, and providing some basic "behaviors" for the robot.

The behaviors built into the robot provide you with some basic actions and responses. When you first power up the robot, it will move forward and turn with a simple preprogrammed action. What makes this a "behavior" is what happens if the sensors detect an object in front of the robot or to the side. In these cases, the robot will stop, back up, and turn away from the object before resuming its random movement. Behaviors are a very powerful feature of robot programming and one that I will discuss in great detail in later chapters.

Even with the remote control and the built-in behaviors, the *TAB Electronics Build Your Own Robot Kit* is not much more than a toy. By adding a Parallax BASIC Stamp 2 (BS2) to the robot, you have a tool that you can use to learn more about programming, electronics, and robotics. I am very excited about this feature (along with its companion "AppMod" socket) and with what different people will come up with on their own.

I am very pleased that Parallax is offering $15 off on a BS2 that can be used with the robot. I strongly urge you to order a BS2 and work through the "Introduction to Programming," "Behaviors and Artificial Intelligence," and "BASIC Stamp 2 Robot Programming" chapters to learn more about robots and learn how to exploit the capabilities of the *TAB Electronics Build Your Own Robot Kit*.

The last point I would like to make in this introduction is that the robot was designed to enable somebody with minimal prerequisite knowledge to program the robot and add electronics devices to it. Programming the robot using a BS2 is very simple; with the ease of developing applications and downloading them into a BS2, I think you will discover that you will be developing your own applications in very short order, no matter what your current skill level.

Bill of Materials

The following parts are included in the *TAB Electronics Build Your Own Robot Kit*:

Quantity	Description
1	Robot PCB with electronic parts attached
1	Infrared remote control
2	Permanent magnet DC motors
2	"Red" wire assemblies
2	"Black" wire assemblies

Quantity	Description
1	Sheet metal motor bracket
2	Sheet metal axle brackets
2	Rubber wheels
2	Plastic Spur Gears, each 0.667" in diameter
2	Axel shafts
2	Plastic worm gears
7	4-40, 1/4" long slot head screws
7	4-40 nuts
7	Lock washers
1	6-32, 1/4" long slot head screw
2	2-56, 1/8" long slot head screws
1	6-32 Acorn nut

Assembly Instructions

Step 1. Press the "spur" gears onto the axles. Next, thread the gear/axel assemblies into the two wheel brackets and then press on the wheels to hold the axle in place. When the axles are put on the wheel brackets, make sure the assemblies are mirror images of each other as is shown in Figure 1-4. The axle and wheel assemblies will be placed on opposite sides of the robot with the wheels on the outside of the robot.

Step 2. For each of the two motors, press the worm gears onto the motor's shaft until the motor shaft is flush with the end of the gear as is shown in Figure 1-5.

Step 3. Mount the motors into the motor brackets and screw the motors to the mount using a 2-56 ⅛" long screw for each motor. The motors should be mounted with the flat side of the axle housing facing upwards and the " + " sign on the motors to the left side. This orientation shown in Figure 1-6 is critical for correct operation of the robot's motors.

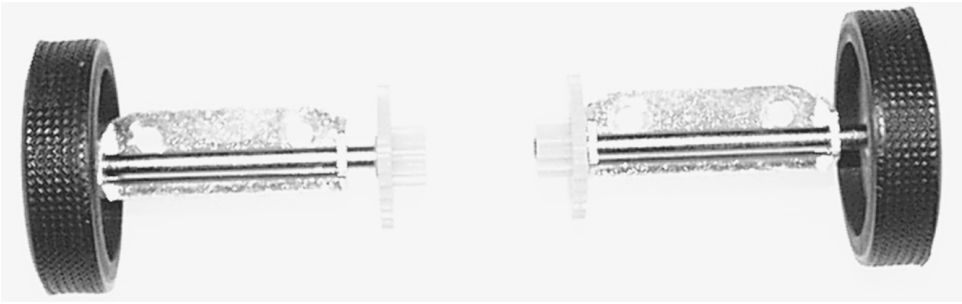

Figure 1-4 Wheels pressed on wheel brackets with axles/gears.

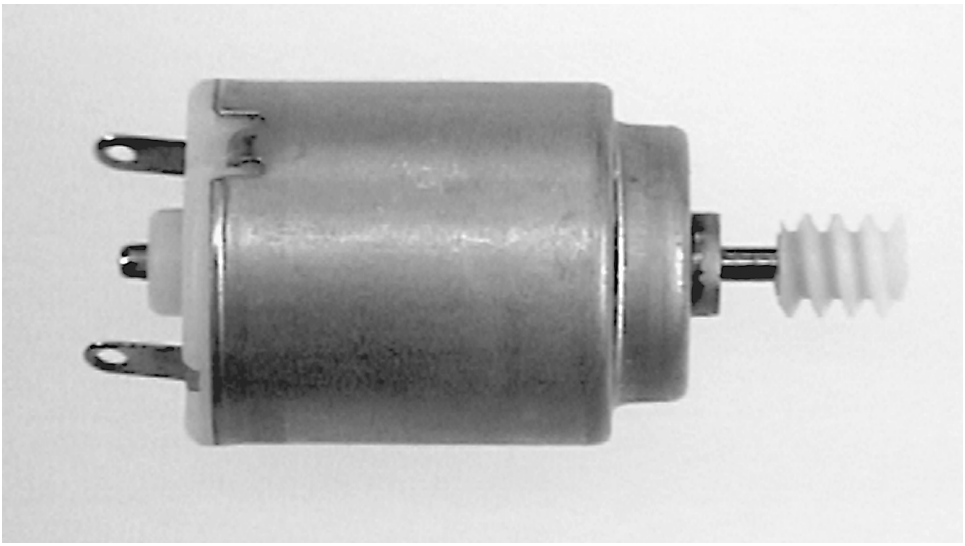

Figure 1-5 Worm gears pressed onto motor shafts.

Figure 1-6 Motors installed in mounting bracket.

Step 4. Plug the red wire assemblies into the two motor power tabs marked "+" and the black wires into the other motor power tabs as shown in Figure 1-7. Looking at the motors from above (Figure 1-7), the red wires should be at the "top" while the black wires are at the "bottom."

Step 5. Bolt the axle brackets onto the rear end of the robot as shown in Figure 1-8, using the 4-40, ¼" long screws, lock washers, and nuts. Note that the screws are behind the axles at the end of the robot. If you have assembled the two wheel/gear/bracket assemblies identically, pull the wheel off one and reverse the bracket before installing that bracket onto the robot.

When installing the screws for the wheel and motor brackets, install with the "heads" of the screw facing the top of the robot, and screw them down to a lock washer and nut on the bottom of the robot as shown in Figure 1-9. It is very important to use the lock washers when installing the screws and nuts, since the vibration of the robot running across the floor will loosen unprotected nuts after a short period of time.

Step 6. Mount the motor bracket to the robot, as shown in Figure 1-10. After the motor bracket is screwed down (using lock washers and nuts as discussed in the previous step), connect the red wire assemblies to the posts marked "RED" on the robot, and then connect the black wire assemblies to the posts marked "BLACK."

Step 7. Install the front "caster." This is accomplished first by putting the 6-32 1/4" long slot screw that comes with the kit through the hole in the front of the robot marked "CASTER" and then into the white nylon "acorn nut," as

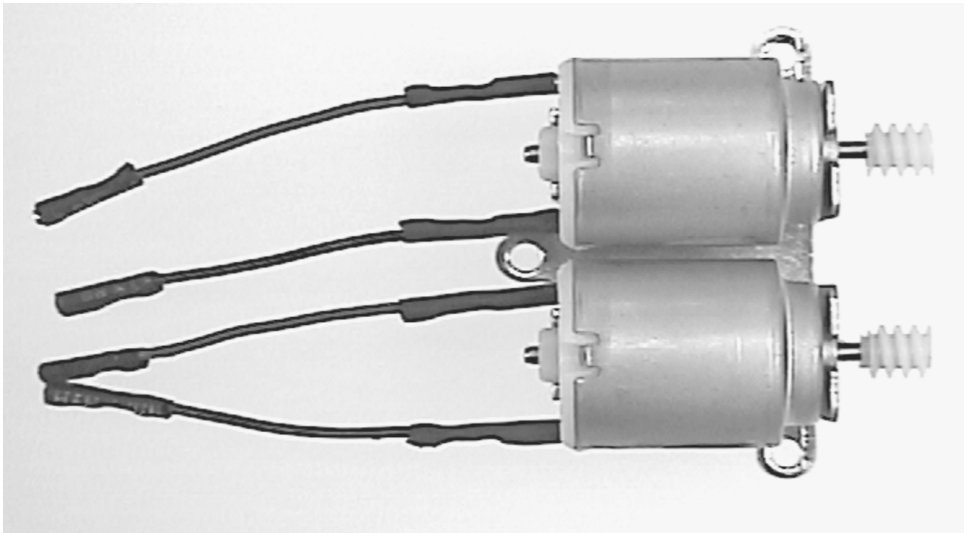

Figure 1-7 Motor wires connected.

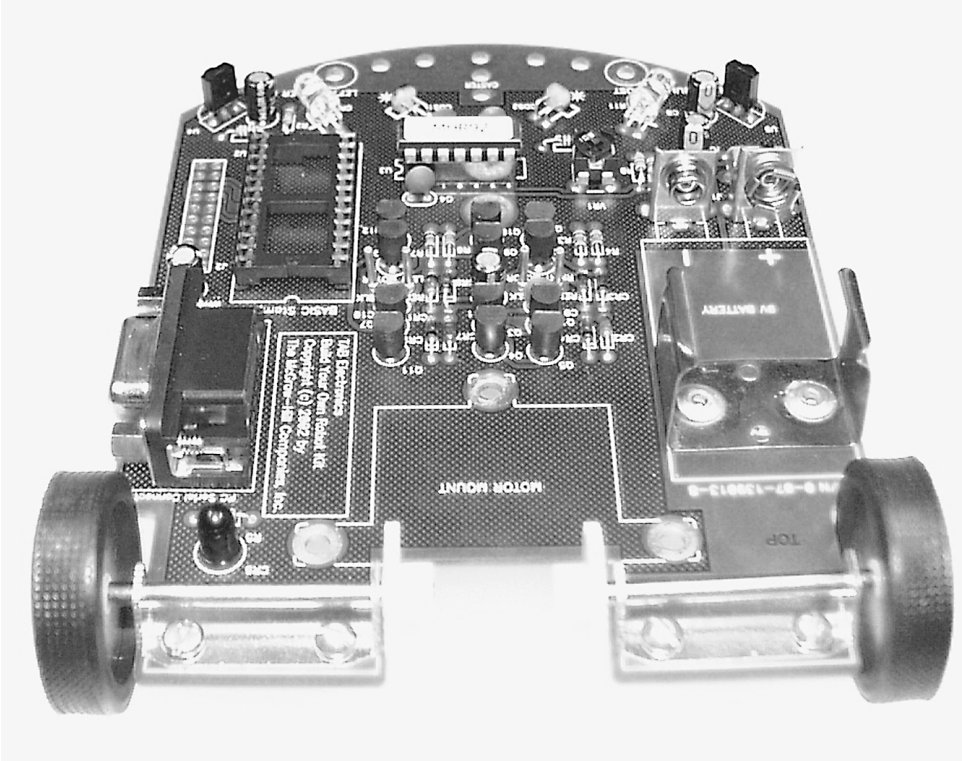

Figure 1-8 Wheel brackets mounted to the robot.

shown in Figure 1-11. The front caster supports the robot and keeps the bottom from scraping against the floor.

Step 8. Install the batteries into the robot and the remote control. The remote control battery compartment has a door on the back that can be removed to install two "AAA" batteries, as shown in the diagram at the bottom of the compartment.

Before installing the 9-volt battery, make sure the robot's power switch is "off." The switch itself (which is just ahead of the battery terminal connectors) should have its "slider" moved to the outside of the robot. After power is applied, the robot will begin to move forward (or in reverse depending on where your hand is) in 1½ seconds. To prevent the robot from jumping out of your hand after installing the battery, make sure the power is switched off.

A 9-volt premium alkaline or nickel-metal hydride (NiMH) battery is installed in the battery clips provided on the robot (marked 9V battery). To install the battery, first place it in the clips that grip the battery from the sides; then push it into the two terminal connectors until the battery "snaps" in. The battery is removed by reversing the procedure—pull the battery away from the terminal connectors, and pull it out from the clips.

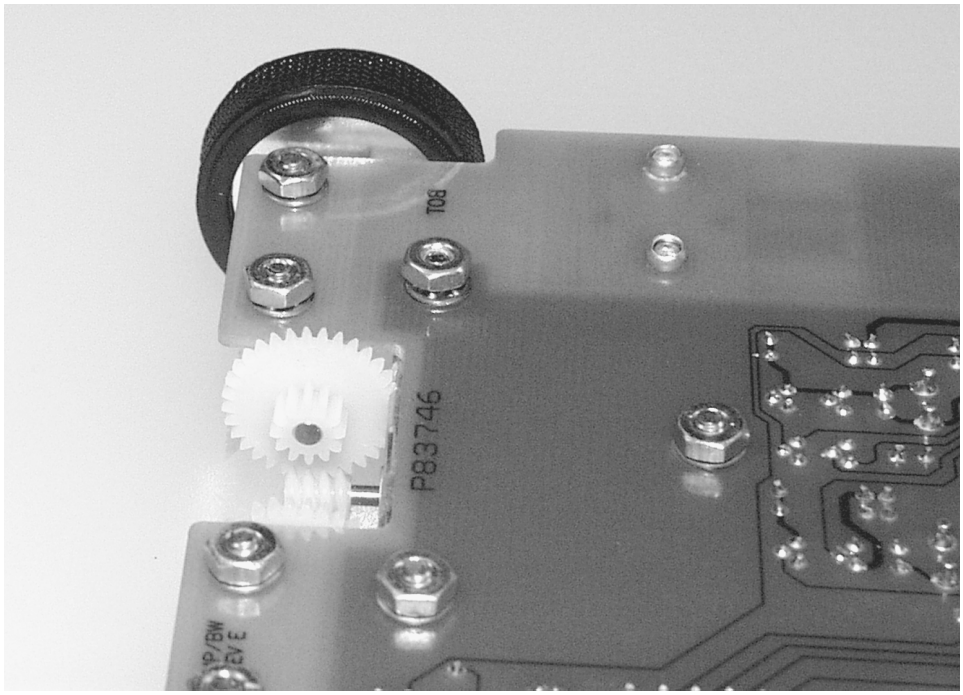

Figure 1-9 Bottom side of robot showing nuts with lock washers in place.

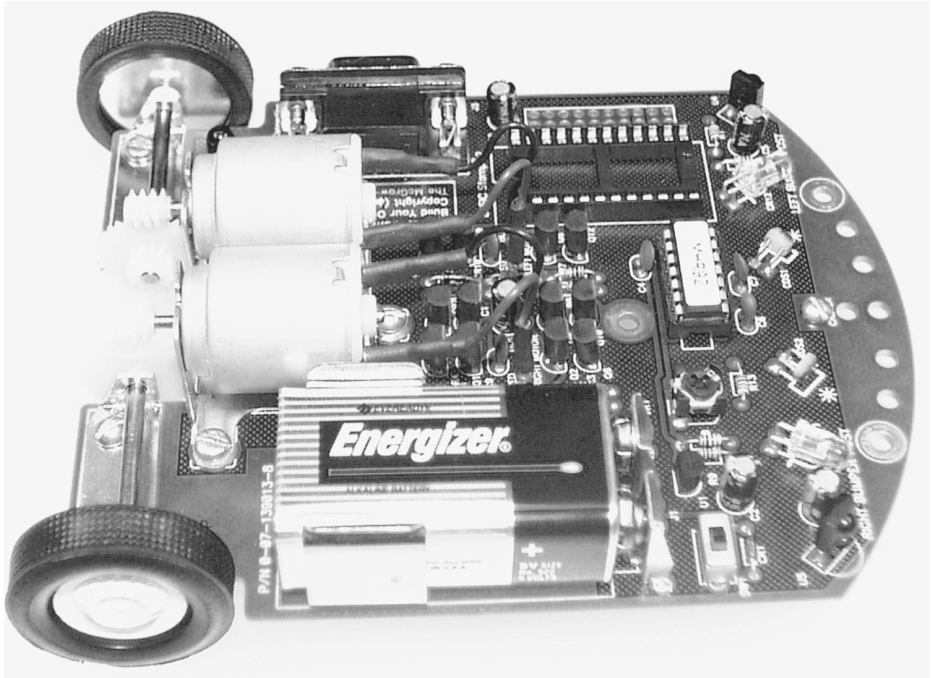

Figure 1-10 Motors installed and wired into the robot.

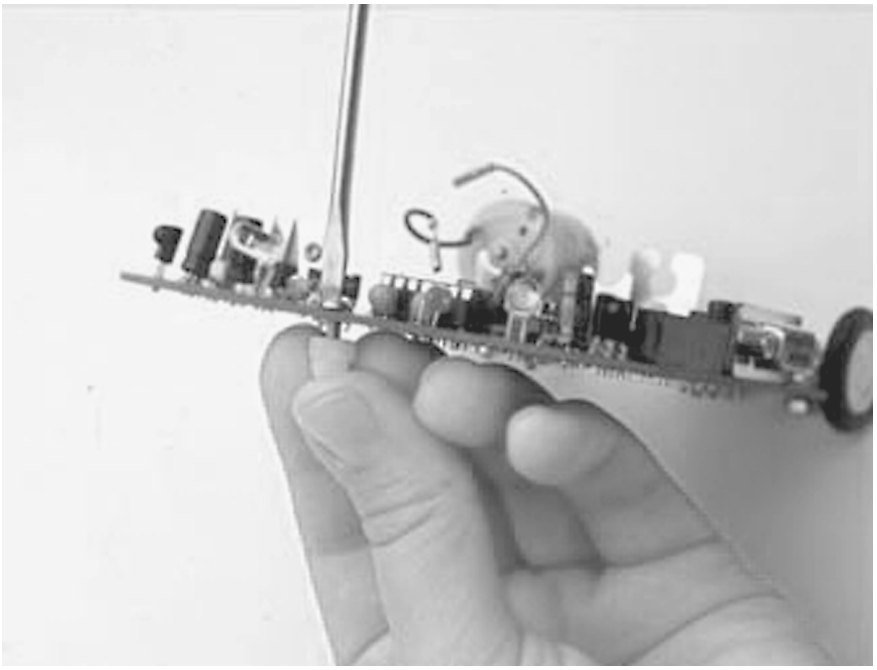

Figure 1-11 Installing the front caster.

Testing and Basic Problem Resolution

With the robot assembled and batteries installed, you are now ready to use the robot.

Place the robot on a smooth, hard surface like a concrete or hardwood floor. Next, move the power switch toward the center of the robot and take your hand away.

After a moment, the LED at the rear of the robot will turn on for half a second and then the robot will move forward as it starts to execute "Behavior 1." This behavior is a random motion behavior that first moves forward for two seconds and then turns the robot for ¾ seconds before repeating. If there is an obstacle in its path, the robot will reverse, turn away from the obstacle, and resume the random movement behavior.

You can stop this (or any of the other behaviors) at any time by pressing the "Stop" button on the remote control.

The robot can be controlled directly by pressing the "Forward," "Reverse," "Turn Left," or "Turn Right" buttons. You can also change the speed of the motors by pressing the "Robot Speed Up" and "Robot Speed Down" buttons on the remote control.

Before going any further, it is a good idea for you to "calibrate" the proximity sensors built into the robot. Follow the procedure outlined in the section, "Calibrating the Proximity Sensors."

Now you can explore the different "behaviors" built into the robot. When the robot is executing "Behavior 2," it will seek out light (this behavior is known as "Photovore"). The robot will move away from light and find the darkest part of the room when the "Photophobe" behavior (Behavior 3) is executing. The final behavior is the "maze solver" or "wall-hugging mode," which causes the robot to follow the wall to its right—this mode can be used for solving mazes or running around the perimeter of the room.

Chances are you are going to have some problems as you work with the *TAB Electronics Build Your Own Robot Kit*. The following table is based on my experiences with a number of prototypes and should give you some ideas on how to fix problems with your robot. You should also check the

`http://www.tabrobotkit.com`

web site for additions to this table as well as use the forums to see if someone has already experienced (and resolved) the same problem that you have.

Problem	Possible Cause	Actions
Robot does not move, and LED does not light on Power Up.	Bad 9-V battery connection	1. Check battery installation.
	Low power 9-V battery	1. Check battery power level. 2. Replace battery.
	BASIC Stamp 2 installed and and commanding "Stop" to robot	1. Check BS2 programming. 2. Program BS2 with "Robot Null 2."
	Both motors jammed/stalled	1. Look at actions for "Motor jammed/stalled."
Robot works erratically and may start reversing unpredictably in Behavior 1.	Low power 9-V battery	1. Check battery power level. 2. Replace battery.
	Proximity detectors not properly calibrated	1. Perform calibration procedure for the proximity sensors. 2. Bend the IR LEDs upward, 5 to 10 degrees, to make reflections from the floor less likely.
Motor stalled/jammed —does not turn.	Motor wiring	1. Check that the "Red" and "Black" wire assemblies are connected to the robot and the motor. 2. Check for internal continuity of the wire assemblies.
	Gears/axles/wheels jammed with lint/hair	1. Look for fibers wrapped around moving parts of the robot drivetrain.
	Dust/dirt on wheels	1. Wipe wheels with a damp rag/paper towel.

Problem	Possible Cause	Actions
Motor runs in wrong direction	Motor wiring is incorrect	1. Check that "Red" wire assembly is connected to both the "+" tab of the motor and the "RED" post on the robot. The "Black" wire assembly should be connected to the unmarked tab on the motor and the "BLACK" post on the robot. 2. Make sure that the motor is installed with the "flat" side of the axle housing pointing away from the motor bracket. 3. Make sure the two motor wire assemblies are connected to the two posts directly in front of the motor.
Robot does not respond to remote control	Low "AAA" batteries in remote control	1. Check battery power. 2. Replace batteries.
	Low 9-V battery in robot	1. Check battery power. 2. Replace battery.
	Remote control button jammed	1. Follow actions for next problem: button continually pressed.
	Remote control's IR LED (at the front of the remote control) is blocked.	1. Point the remote control directly at front of robot. Make sure nothing is blocking a clear LED at the front of the remote control.
Robot behaves as if remote control button is continually pressed.	Check for button jammed down	1. Inspect each button to see if a button is caught under the plastic legend plate. 2. Press each button on the remote control to see if you can release the button.
	Low 9-V battery in robot	1. Check battery power. 2. Replace battery.
When in "direct control," the robot runs into a wall without the proximity sensors stopping it.	This is normal operation.	1. It is assumed that if the robot is under direct control, the operator will not allow it to be damaged by colliding with different objects in the room.

If you have behavior problems or problems with a BS2 installed in the robot, please look at the chapters that describe these functions for information on possible problems.

Calibrating the Proximity Sensors

After you have assembled the robot and experimented with controlling it using the remote control, you should take a minute and "calibrate" the proximity sensors. This procedure is very simple and should only take you a few seconds. The only materials you need are the assembled *TAB Electronics Build Your Own Robot Kit* robot, the remote control, a small Phillips screwdriver, and a small piece of paper.

First power up the robot and press "Stop" on the remote control (the button on the remote control that has the "stop sign" icon). This will stop "Behavior 1" from executing and will place the robot in a mode where it is continually polling (checking) the IR proximity detectors.

Place the piece of paper 3 to 4" from the robot's left IR detector and, with the Phillips screwdriver, adjust the potentiometer until the LED at the rear of the robot flashes on and off. This procedure should look like Figure 1-12. When I perform this operation, I normally use a business card. Make sure that the right proximity sensor (IR LED and IR detector) does not have an object near it (like your hand). This could result in a "false" object proximity detection.

When you are calibrating the proximity sensor, there are a few things you should watch for. The first is that the running surface (floor) may be reflecting back the IR signal, causing a false proximity sensor indication. I found that bending the two IR LEDs upward by 5 to 10 degrees prevents the floor from being a problem.

Also, make sure that the IR detectors are not in a direct line of sight to the IR LEDs. You can bend them backwards a few degrees so that they are more in the "shadow" of the capacitors (tall, round components between the IR LED and IR detector).

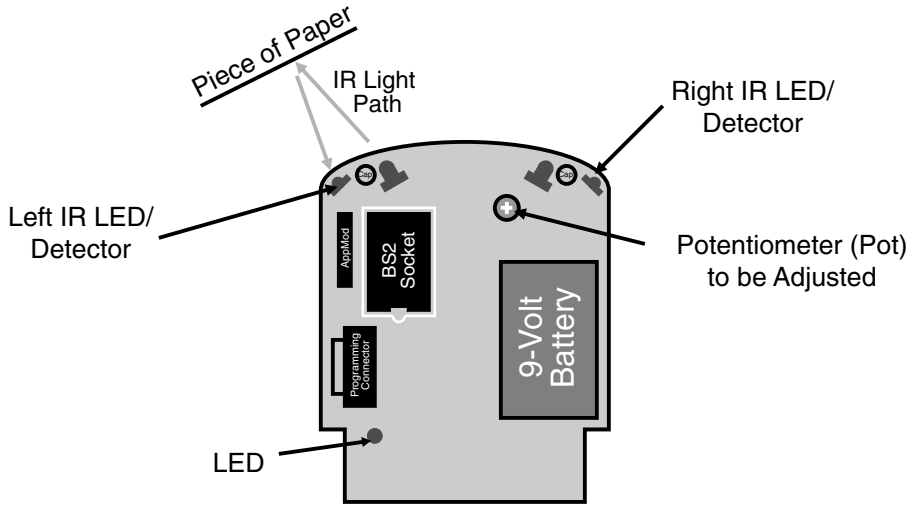

Figure 1-12 Calibrating the robot's proximity sensors.

Remote Control

One of the unique features of the *TAB Electronics Build Your Own Robot Kit* is the infrared (IR) remote control that is used to control it. This device allows you to explicitly control what the robot does and direct the robot to perform one of four basic "behaviors" built into it. In addition to controlling the robot, you can use the remote control to communicate with the Parallax BASIC Stamp 2 that can be optionally used to control the robot.

The remote control has the somewhat unusual button legend shown in Figure 1-13. Instead of describing the functions of the buttons, small icons are used to represent the different responses of the robot or to represent changes within the robot's operating parameters. In Figure 1-13, I have labeled and identified each button, and I will expand on the function of each direct control button in this chapter as well as in the following chapters.

In Figure 1-13, I have labeled three buttons (each with a number of "bars" beneath it) as "BS2 Control Buttons."

The remote control requires three AAA batteries. The power required by the remote control is very modest, and it will work with virtually any batteries—it does not require the high-performance alkaline or rechargeable batteries that the robot has.

When the remote control sends a command to the robot, you will see the LED at the back of the robot flash as it receives each data "packet." This function is disabled any time the BS2 sends a command to the robot. I mention this because one of the BS2 applications I provide in a later chapter simply sends a "Stop" command to the robot and does nothing else (allowing you to control the robot by the remote control). This can be confusing because after this application executes, the LED no longer flashes when a packet is received from the remote control.

In the following sections, I will describe the functions provided by the remote control and how they can be used with the robot.

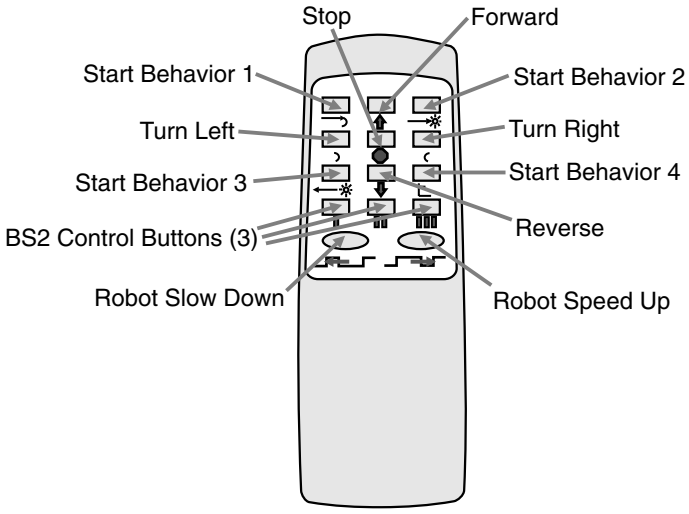

Figure 1-13 IR remote control layout.

Direct Robot Control

Six buttons on the remote control enable you to manually move the robot and specify the speed. These buttons and the action of the robot caused by pressing a button may work the same way as the other behavior and robot buttons, but I believe their operation is quite intuitive.

To move the robot, the "Forward" and "Reverse" buttons are used. When these buttons are pressed, both wheels are turned in the same direction and have the same voltage and pulse-width-modulated (PWM) signal applied to them. I mention this because you can perform an interesting experiment as soon as you have the robot assembled and batteries installed.

Place the robot at one end of a room with its back against the wall and press the "Forward" button until the robot reaches another wall. I say "another" wall because you will find that the robot will veer off course as it goes across the room. The cause of the turn can be an unequal number of windings in each of the two motors, some dust on the floor causing a wheel to slip, or a difference in the actual diameter of a wheel or gear compared to its opposite one. I have built almost a dozen prototypes of the robots and have used a variety of different motors with them, and I have seen them turn both left and right and occasionally go straight ahead.

This is, no doubt, the major problem with the differentially driven robot—without some kind of positive steering, it's just about impossible to turn the wheels at the same rate. Even if you could, you would never be sure whether or not the wheels had slipped. Some differentially driven robots use stepper motors to run the wheels at the same speed (and to move an equal distance), but because of factors such as wheel slip, you still can never be sure of the robot's position.

Turning the robot is accomplished by pressing the "Turn Left" and "Turn Right" buttons. When the robot turns, it turns about a point that is between the two wheels and is not central to the position of the robot. The turning action is shown in Figure 1-14.

Depending on the running surface of the *TAB Electronics Build Your Own Robot Kit*, you may see one wheel spinning, causing the turn to look more like a "skid" (or a "doughnut"). This is another manifestation of the problem I described above with "Forward" and "Reverse."

When you press the "Forward," "Reverse," "Turn Left," or "Turn Right" buttons, the robot will move in the command manner until the button is released. The remote control sends a "repeat packet" once every 100 milliseconds (or 10 times every second), which is used to reset counters within the robot until the button is released. After the button is released, the robot continues the command for 200 milliseconds as the counters count down to zero.

Based on the above, you may feel that the "Stop" button is redundant or not needed. The "Stop" button is used primarily to stop the four behaviors built into the robot and provides a command that initiates the IR proximity sensor "calibration."

The speed at which the robot moves can be changed by pressing either of the oval "Speed Up" and "Speed Down" buttons. As I have indicated above, there are four speed ranges plus "Stop."

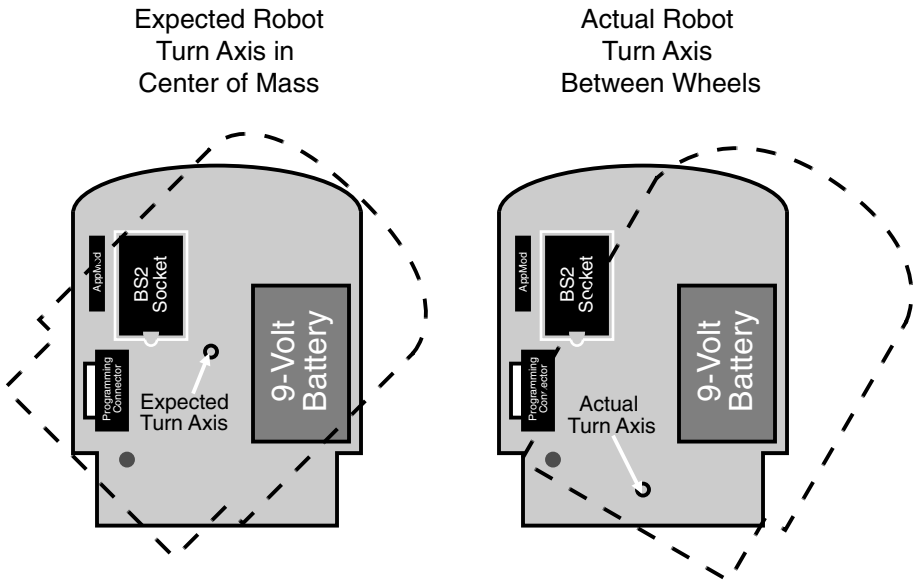

Figure 1-14 Actual versus expected robot turn axis.

The natural expectation would be for the robot to "throttle" to 25, 50, 75, and 100%, but this is not the case for this robot (and indeed for many other pulse-width-modulated controlled electrical devices). This is because a certain amount of power is required to turn the motor and move the robot. I will explain this in more detail in later chapters, but for your reference right now, the different power level "notches" are:

Notch	Power level
4	100%—Full Power
3	75%
2	56%
1	40%
0	0%—Full Stop

These power levels are not simple calculations based on the time power is applied to the motors. In the following chapters, I will describe what PWM motor control is and how it is applied with the *TAB Electronics Build Your Own Robot Kit*.

Demonstrating the Robot's "Behaviors"

I suggest that you spend some time experimenting with and learning about the four different "behaviors" built into the robot. Robot behaviors are an extremely powerful programming concept that you can take advantage of when you develop your own applications for the robot. In later chapters, I will explain how the behaviors have been implemented as well as how they can be accessed directly by a Parallax BS2.

"Behavior 1" is the default behavior of the robot and consists of a random motion, moving forward for two seconds, and then turning for three quarters of a second, backing away and turning away if it collides with an object. You may think that this behavior is useful just as a robot "demonstration mode"—but it can be very useful in actual robot applications to move the robot to a random place on the floor. As the battery charge changes, you will see the speed of the robot executing this behavior change, which results in the final position of the behavior to be always different.

In summer months, I'm sure you've noticed that if you leave an outside light on it will have a multitude of insects circling around it. Virtually all animals in nature will leave dark areas for lighted ones. The *TAB Electronics Build Your Own Robot Kit*'s "Behavior 2" simulates this behavior by checking the light levels at each of the two CDS cells and turning toward the brightest. When the robot encounters an obstacle, it stops (and doesn't circle like an insect).

Historically, this behavior is known as a "photovore," which translates literally to "light eater." The first robot experiments implemented this behavior to see how a machine could simulate the behavior of an insect attracted to a light.

To demonstrate the photovore behavior, I recommend that you place the robot in a dark room with a flashlight as shown in Figure 1-15. If you were to place the robot in a normally lighted room, you would see the robot move about in a seemingly random manner, since it is unlikely that there is a single point brighter than all the others. In addition, you may discover that the robot will encounter local areas of "brightness" in which the robot cannot "see" its way clear.

Depending on the room used for testing the robot, you may find that it is attracted to light reflected off baseboards and furniture instead of the primary light source. This problem is caused by the robot's relatively small angle of "sight" along with its inability to determine the source of light.

The robot stops Behaviors 2 and 3 when an obstacle is detected by the IR proximity sensors.

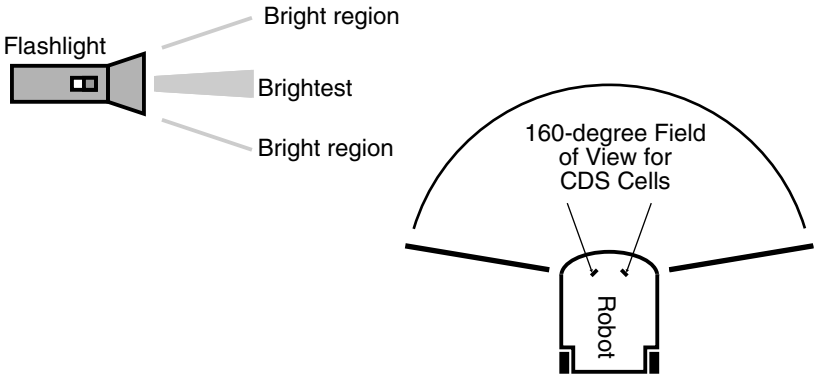

Figure 1-15 Apparatus for testing Behavior 2.

The opposite behavior to the "photovore" is obviously moving away from light and searching out the darkest point in a room. This is built into the *TAB Electronics Build Your Own Robot Kit* as "Behavior 3." Roboticists call this behavior "photophobe," or "afraid of the light."

While I suggest that the photovore be used in a darkened room with a single light source, the photophobe behavior is very interesting to observe in a normally lighted room. As I will discuss in later chapters, the "perspective" of the photophobe is continually changing, and you will find that the robot will travel across half the room only to turn away and resume searching for the darkest spot in the room.

"Behavior 4" is different from the previous three behaviors, since it causes the robot to stay close to objects and obstacles. Objects and obstacles are regarded as a "bad thing" in the other three behaviors, with the robot either backing away from objects or stopping altogether. Behavior 4, known as the "wall-hugging" or "maze-solving" behavior, is designed to keep the robot one or two inches from a wall or object. If the robot moves away from an object, it will then turn back into it. If the robot encounters an obstacle, it will turn away from it and the obstacle will become the new object the robot follows.

To test out this mode, place the right side of the robot (the side the 9-volt battery is connected to) an inch or so away from a wall, as shown in Figure 1-16, after you have pressed "Stop" on the remote control. When you are ready, press the Behavior 4 button (the button with the icon that looks like a partial maze), and watch the robot "jitter" along the wall, turning toward and away according to what the proximity sensors determine to be the correct distance from the wall.

You may find that the robot will collide or bind with objects that turn away to the right. This problem can be solved by simply recalibrating the robot's proximity sensors to indicate a collision further away. While I have found that three inches will work in just about every robot, you may want to start at four inches and work your way back.

Each of these "behaviors" can be invoked when a BASIC Stamp 2 is added to the robot. This will make the task of developing applications quite a bit easier. It will allow you to look at the "big picture" of how the robot application works and avoid getting stuck in the minutia of programming behaviors into your application.

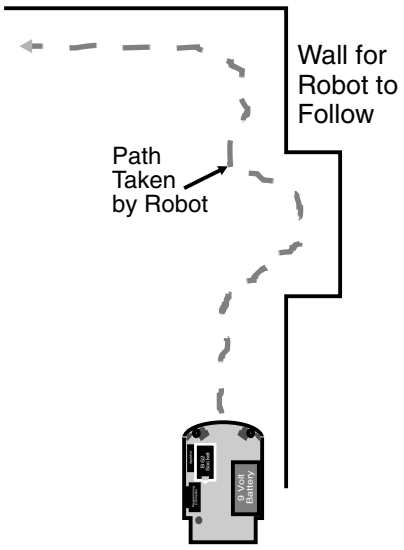

Figure 1-16 Robot in "wall-following" mode.