From: Mobile Robots Inspiration to Implementation (2nd ed), by J.L. Jones and A.M. Flynn, A.K. Peters Ltd., Natick, MA, 1999

2

TuteBot

2.1 A Tutorial Robot

Building a robot can be a lot of work. All the more so, if the first plan is unnecessarily complex. This chapter is intended to help get you started with building robots while also illustrating some key points about designing a robot's intelligence system without becoming too encumbered in the myriad of details involved in creating a more sophisticated creature. We will show just how simple a robot can be and launch you on your way to building one.

Before proceeding to the more sophisticated Rug Warrior described in the next several chapters, we will begin here by constructing TuteBot—a robot which is simple yet complete. Do not underestimate the elegance of simplicity, however. It is often the simplest solution which takes the longest to comprehend, and yet it is also often the simplest solution which illustrates the main lessons with the most clarity. Experienced designers of robotics and automation systems agree that the first way they design something is usually the most complex way. Difficulty usually arises when trying to simplify the system.

TuteBot will exemplify how a robot as a system, a collection of sensors, actuators and computational elements, can be organized in

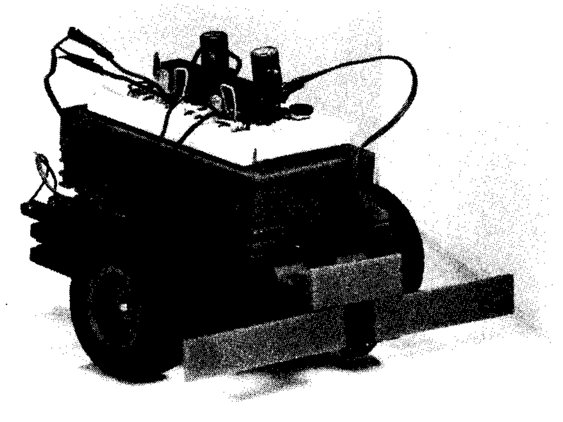


Figure 2.1. TuteBot is a robot that can explore its surrounding, escape from collisions with obstacles, and be programmed to follow walls.

such a way that intelligent actions result in response to certain stimuli. TuteBot will consist of a circuit, a chassis, a sensor, a battery, and two motors. It can be programmed by adjusting two potentiometers. The entire robot will be built from Fischer-Technik parts and a few electronic components which are readily available from Radio Shack and other electronic stores.

What will the TuteBot be able to do? Its repertoire of behaviors will endow it with the capabilities to explore its world, escape from objects with which it collides, and follow along walls that it detects with its bumper.

A completed TuteBot is shown in Figure 2.1. The front bumper acts as a sensor and detects collisions with obstacles in its path. A trailing caster wheel maintains stability. Above the chassis is the battery case and mounted on top of the batteries is the breadboard containing TuteBot's electronic circuitry.

All the mechanical components used here are Fischer-Technik parts: motors, gears, axles, wheels, switches and connectors. Fischer-Technik is an excellent source of parts for building robots as the designer can prototype mechanisms quickly without recourse to a machine shop. The Fischer-Technik parts and pieces are available in multiple quantities. Catalogs are available and should be requested.

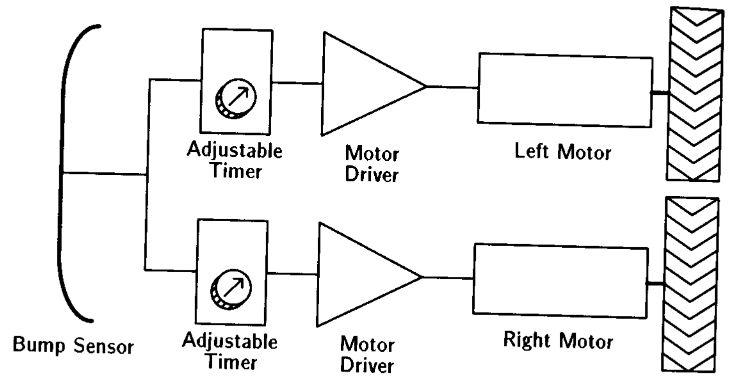


Figure 2.2. The essence of TuteBot. Two motors, two wheels, a bump sensor, two potentiometers for programming, and two motor drivers are enough to create a concrete example of a simple robot—an intelligent connection of perception to action.

An address and phone number for Fischer-Technik is listed in Appendix C. Other types of mechanical building block kits are also quite usable and widely available, including LEGO, LEGO Technik and Meccano.

TuteBot's brain is entirely analog circuitry. No integrated circuits are required and almost all of the components, including the breadboard, can be found at a Radio Shack store. The only tools required to put TuteBot together are wire cutters, wire strippers, and possibly a soldering iron for making connectors. An oscilloscope is not necessary although having one always makes debugging easier. A multimeter should suffice for debugging TuteBot.

A block diagram of TuteBot, shown in Figure 2.2, illustrates how the bump sensor is connected to the actuators. The signal created when the bump sensor detects contact is sent to the motor-driver circuitry for each wheel, signaling the robot to back up. Adjustable timers associated with each motor driver determine for how long each wheel should reverse.

2.2 TuteBot Behaviors

With a minimal amount of hardware, obstacle avoidance can be implemented on TuteBot. Figure 2.3 depicts the sequence of actions

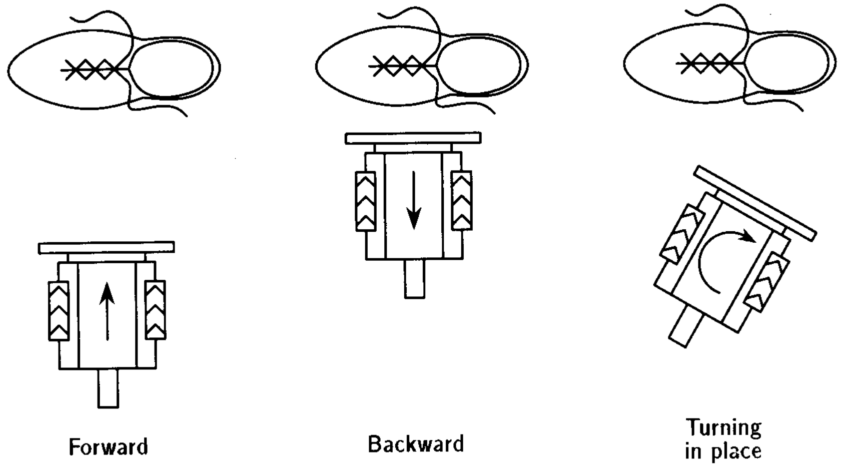


Figure 2.3. TuteBot's basic operation. When TuteBot is powered up, the robot moves forward until it encounters an obstacle. TuteBot then backs up, turns in place, and resumes its forward motion. The time spent backing up and turning in place is programmed by the user.

that occur when TuteBot strikes an obstacle. The robot is initially moving directly forward toward the shoe. As it strikes the shoe, both motors are switched to reverse and the robot backs straight up. However, one motor stays in reverse longer than the other and the robot begins to turn; in this case, the right motor reverses for a longer time period, causing TuteBot to turn to the right. At some point, the right motor stops reversing and both motors go forward, leading TuteBot off in a new direction, hopefully with a wide enough berth to avoid the shoe. If not, the TuteBot bumps into the shoe again and the process repeats until TuteBot turns far enough to the right to finally avoid the shoe.

A timing diagram which graphs this sequence of events is shown in Figure 2.4. The top graph depicts the signal generated by the front bumper's bump sensor. The bottom two graphs illustrate the signals sent to the right and left drive motors.

Initially, both motors receive signals which direct them to go forward. If a collision occurs, the bumper sends a binary signal to the adjustable timers—low for no-contact, high when an obstacle is struck. The timers, in turn, provide a binary signal to the motor

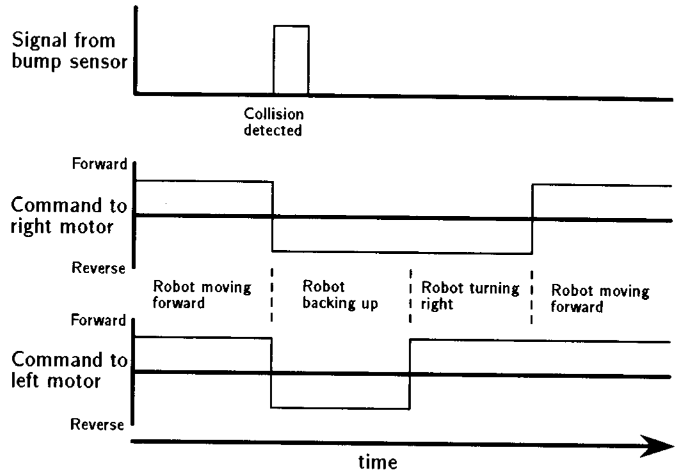


Figure 2.4. The timing sequence generating TuteBot's backup behavior. Both motors normally move in the forward direction as shown in the bottom two graphs. When the bump sensor is activated, both motors reverse. The right motor continues in reverse longer than the left, causing TuteBot to turn to the right. When both motors resume forward motion, TuteBot moves on in a new direction.

drivers—high for forward rotation, low for reverse rotation. Once activated, each timer continues to supply the low signal for a characteristic time. The motor drivers interpret this high or low signal by providing forward or reverse current to the motors respectively.

Assume that the timers are set for delays of t_r seconds and t_l seconds for the right and left motors and that $t_r > t_l$. After encountering an obstacle, the robot will backup for a time t_l. It will then turn to the right (the left motor turns forward, the right motor stays in reverse) for a time $t_r - t_l$. It will then resume moving forward in a different direction, thus avoiding the obstacle or repeating the sequence until it does avoid the obstacle.

An additional behavior can be made to emerge from the robot. If we bias the motors so that, when going forward, one motor turns faster than the other, the robot will move in an arc. This slowdown in speed can be implemented by adding a resistor in series with one motor. If, for instance, the left motor is forced to turn significantly more slowly than the right, the robot will arc to the left. By combining this forward arcing behavior with the earlier back-and-turn behavior, TuteBot can be coerced to follow a wall as was illustrated in Figure 1.2.

To demonstrate this, one would place the robot with a wall to its left, and adjust the timers so that, after encountering a bump, the robot backs and turns a bit to the right. Now when going forward, the robot arcs to the left until it hits the wall; then it backs up, turns right, and then heads forward in an arc until it bumps the wall again. For suitable settings of the parameters, it should be able to turn through a doorway and negotiate either inside or outside corners.

It is an important point here, that nowhere in TuteBot's simple brain does it have knowledge of what a wall is or what is required to follow a wall. Rather, the superposition of a simple set of reflex actions allows a more complex behavior to emerge. This idea of seemingly complex behaviors emerging from a collection of simple rules is the underlying notion of behavior control, which we introduced earlier. We will see more complex examples when we get to the microprocessor-controlled Rug Warrior.

2.3 Building TuteBot

TuteBot senses the world through a front bumper. It steers by individually changing the direction of its drive wheels, while a trailing caster wheel supports the robot. A simple relay, transistor, and capacitor circuit provide all the computational power TuteBot needs.

Figure 2.5 lists the parts needed to construct TuteBot. Most of the parts are available from Fischer-Technik. The remaining parts are easily obtained at Radio Shack or another electronics store. We will begin describing the construction of TuteBot by stepping through the mechanical layout of how to mount the motors and attach the wheels.

Motors for TuteBot

The Fischer-Technik motors have an attached worm gear, transfer box and large axle-mounted gear. Direct current (DC) motors usually spin too fast and have too little torque to drive the loads of the wheels. "Gearing down" a motor causes a motor to spin more slowly

DESCRIPTION	F/T PART NO.	QUANTITY	AREA USED
BOTTOM HALF OF HINGE	31426	2	BUMPER
TOP HALF OF HINGE	31436	2	BUMPER
4W 4P/F	38464	2	BUMPER
2W 2P/F	38242	1	BUMPER
CASTER WHEEL PIVOTHOLDER	32321	1	CASTER WHEEL
CASTER WHEEL PIVOT	31124	1	CASTER WHEEL
CASTER WHEEL WHEELBLOCK	32085	1	CASTER WHEEL
CASTER WHEEL AXLE	31690	1	CASTER WHEEL
CASTER WHEEL TIRE	36573	1	CASTER WHEEL
1W P/C	37237	2	FRONT END ASSEMBLY
1W2L P/C 4C	32879	2	FRONT END ASSEMBLY
1W1L P/5C	32881	4	FRONT END ASSEMBLY(2) AND WHEELS(2)
LEFT AND RIGHT MOTOR	32293	2	MOTOR
LEFT AND RIGHT GEARBOX	31078	2	MOTOR
1W1L P/P +4C	32882	1	MOTOR
2W C/P	35049	4	POWERBLOCK
1W P/P	37238	2	POWERBLOCK
POWERBLOCK TOP	35986	1	POWERBLOCK
POWERBLOCK BOTTOM	36165	1	POWERBLOCK
WIREHOLDER	35969	1	POWERBLOCK
ANGLE C/P	38423	2	SWITCH
SWITCH	37783	1	SWITCH
GEARHUB	35031	2	WHEELS
GEARHOLDER	35033	2	WHEELS
GEAR	31021	2	WHEELS
BIGWHEEL HUB HOLDER	31058	2	WHEELS
BIGWHEEL HUB	32883	2	WHEELS
1W2L 5C	32880	2	WHEELS
60 mm SHAFT (METAL)	31032	2	WHEELS
TIRES	32913	2	WHEELS
ELECTRIC PLUG (GREEN)	31336	4	WIRING
ELECTRIC PLUG (RED)	31337	4	WIRING
WIRE	31360	2	WIRING

W = WIDE; L = LONG; P = PIP; C= CHANNEL; F = FLAT

Figure 2.5. TuteBot can be constructed from these or similar parts.

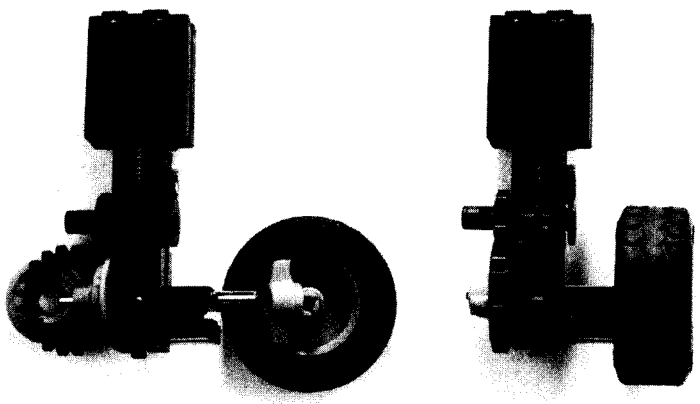


Figure 2.6. Assembly of the motor, transfer box and wheel gear.

but with more torque at the output of the gear stage. Thus, the wheel can push against the floor with more force.

This worm gear, transfer box and axle-mounted gear which is unique to Fischer Technik gives the TuteBot a total speed reduction of about 30:1 and wheel revolutions of about one per second. A speed reduction of between 20:1 and 30:1 is an appropriate goal when using Fischer-Technik or other motors and gears. Gears and motors are explained in more detail in the later chapter on motors.

The first step is to build the left-side motor, transfer box, and wheel and gear assembly as shown in Figure 2.6. (Follow the same steps for building the right-side motor assembly.) The left and right sides are the same except that one of the transfer boxes is upside down so that both 10-tooth gears are facing inward.

The two motor-sides are joined by a connector block as shown in Figure 2.7(a). Notice that this connector block is actually made from three Fischer-Technik pieces (2.7(b)).

The caster wheel which is the rear wheel for the TuteBot is assembled from five Fischer-Technik pieces. They are shown in Figure 2.8 both apart and assembled. This caster wheel assembly fits in the back of the chassis between the two motors. It should slide in between them and allow the chassis to stand on three wheels.

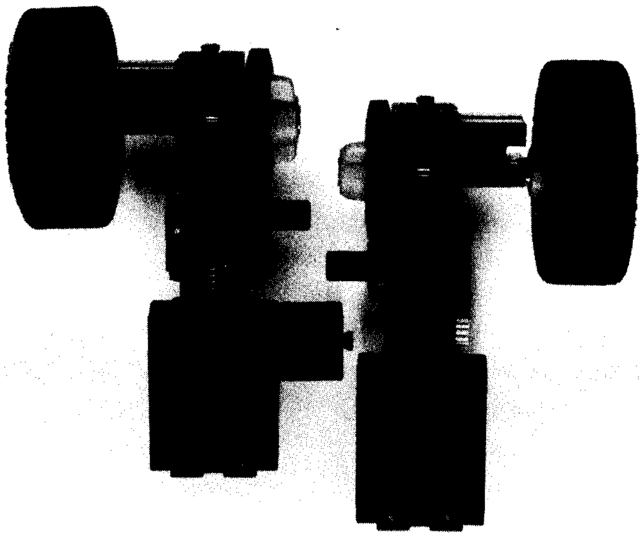


Figure 2.7. Joining the motors with a connector block.

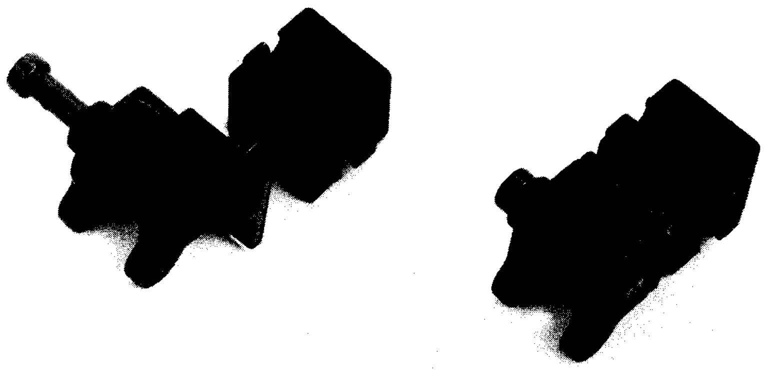


Figure 2.8. Assembling the caster wheel.

The TuteBot chassis can be constructed by following the sequence of steps outlined in Figure 2.9.

Front-end assembly

The front end of the TuteBot is made from four black pieces and two red pieces. Slide one of the black double-long blocks into the other and slide two black single blocks onto either end, half the distance of each block. Then, holding the joined blocks with the two single blocks facing up, slide the two red pieces onto the same side of the joined double blocks, as shown in Figure 2.9(a). This side is now the back of the front-end assembly.

Switch assembly

The switch slides into two red L-pieces; this switch-assembly slides onto the front side of the front-end assembly just constructed. (See Figure 2.9 (b))

Bumper assembly

The actual bumper is composed of seven pieces, shown in Figure 2.9 (c).

The completed bumper assembly slides onto the front side of the two single blocks on the top of the front-end assembly.

To complete the chassis slide this entire unit onto the front of the wheel gear-motor assembly.

Connecting power to the chassis

Now we will connect the power block (battery pack) to the TuteBot chassis. The power block requires six AA alkaline batteries. Attach the wire-holder connector to the front right-hand side of the power block. (The power holes are on the back side). Using one short and one long red piece each (see Figure 2.10(a) construct two "legs" and attach them to the rear of the power block. Slide the power-block assembly onto the motors at the rear of the chassis. The front of the power block should simply rest on the front-end assembly (see Figure 2.10(b)).

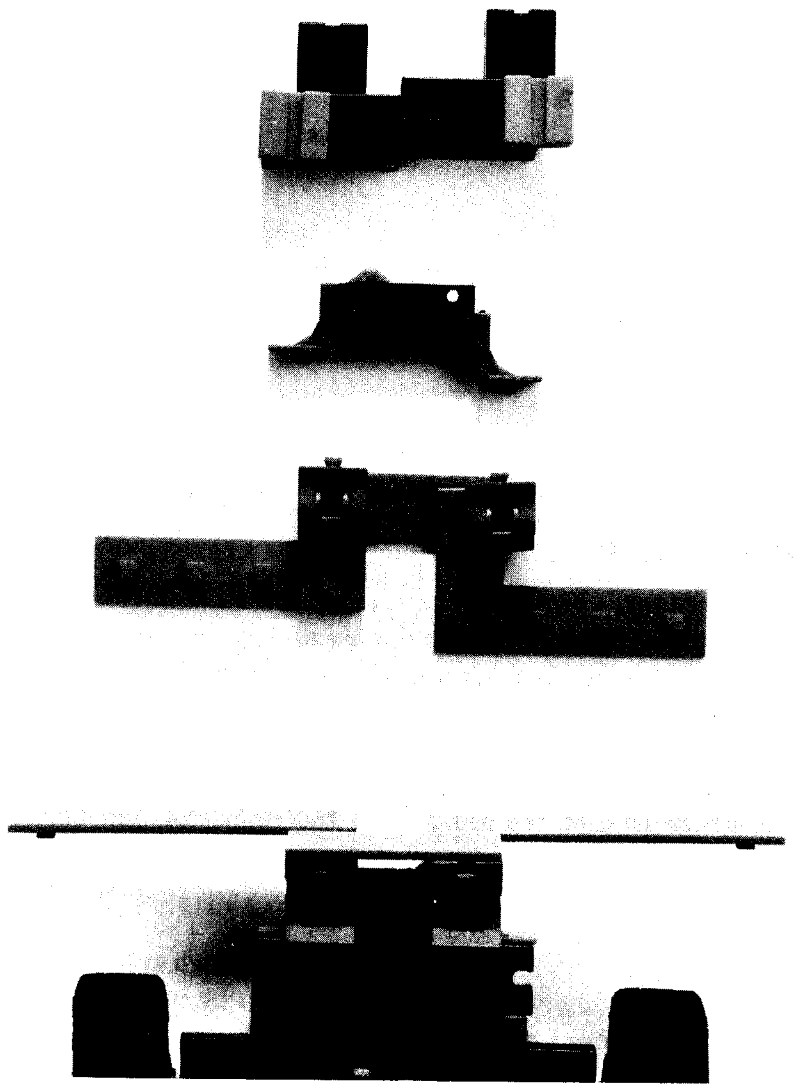
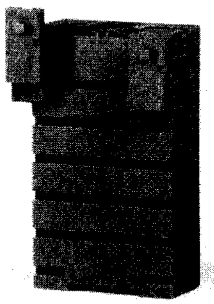


Figure 2.9. Step-by-step instructions for building the TuteBot chassis. (a) Front-end assembly. (b) switch assembly (c) bumper assembly (d)connection of bumper assembly to front-end assembly.



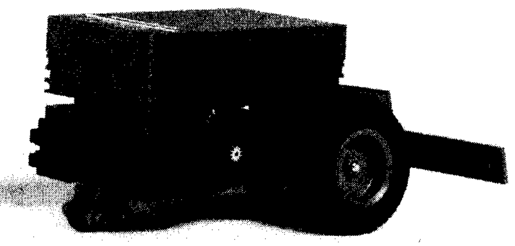


Figure 2.10. (a) Constructing "legs" for the power block. (b) Attaching the power-block assembly to the motors.

Wire Connectors

Next, we will make connectors for the motors and bump switch. Two Fischer-Technik connectors should make all four required connectors for TuteBot. Fischer-Technik provides connectors that fit with their motors and switches. However, the other ends of these cables must be modified so that they can be plugged into TuteBot's breadboard. Cut a Fischer-Technik connector in half and connect two inches of red 22-gauge wire to the red wire of the connector and black 22-gauge wire to the green wire of the connector. Make three such connectors. For the fourth connector connect a two-inch green 22-gauge wire to each of the red and green wires. These connections should be protected with electrical tape. Although color makes no difference electrically, you can avoid confusion by using a green 22-gauge wire to connect to the switch.

Plug the wire connector with the attached green wires in the holes labeled 1 and 3 on the switch. This sets the switch up as a normally open (NO). It is this switch that will cause the TuteBot to back up when it collides with an obstacle.

Plug two of the three remaining connectors into the left and right motors at the rear of the chassis.

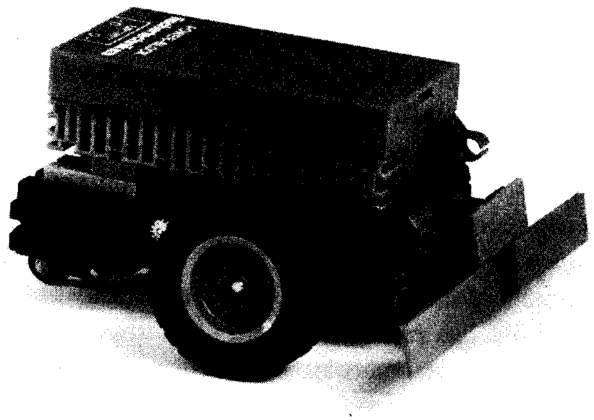


Figure 2.11. The TuteBot is now assembled. All that is left is to mount the breadboard to the top of the battery pack.

Plug the remaining connector (red to +, green to -) into the corresponding holes at the back of the power block.

Now your TuteBot should look similar to that in 2.11. The final step is to mount the breadboard to the top of the battery pack.

In the next section, we will discuss building the electronic circuitry for TuteBot's brain. Once this has been assembled, mounting it on top of the constructed TuteBot should produce a robot resembling that in Figure 2.1, shown at the beginning of this chapter.

2.4 Electronic components

Before we get into the specifics of the control system for TuteBot, we take a moment here to describe the basics of a few common electronic components such as relays, transistors, capacitors, diodes, etc. Figure 2.12 illustrates the relationship between the physical components we will use on TuteBot and their schematic symbols.

First, the *relay* shown in the upper left-hand corner of Figure 2.12 is a type of electrically controllable switch. TuteBot uses relays to switch the polarity of the voltage applied to its motors and thus reverse their direction. The idea behind a relay is that a small current flowing in the relay's coil can allow much larger currents to flow through its contacts. The way a relay works is that, when different voltages are applied to the two lines marked coil, the resulting current creates a magnetic field inside the device. This field attracts a metal lever to which the internal switch contacts are attached. Activation of the lever in turn disconnects one circuit and connects the other. With no voltage applied, the line marked com or *common*, is connected to nc , the *normally closed* pin. When a voltage is applied across the coil, com is disconnected from nc and connected to no, the *normally open* pin.

Next come bipolar *transistors*. Bipolar transistors have three terminals: a *base*, *b*, a *collector*, *c*, and an *emitter*, *e*. For a particular transistor case design, the correspondence between these symbols and the physical leads can be found in the manufacturer's data book. Transistors can be used as amplifiers or switches. TuteBot employs transistors to supply a current sufficient to activate the relay. There are a great variety of transistors. Two of the important parameters that differentiate among them are amplification factor and maximum power-handling ability.

A *diode* is a device which allows current to flow in one direction but not the other. If the "+" end of a diode, the *anode*, is connected to the "+" terminal of a battery and the "-" end of the diode, the *cathode*, is connected to the "-" terminal of the battery, a large current will flow through the diode, enough to damage the diode or battery. Usually a resistor is placed in series with a diode to limit current to a safe level. If the connection is reversed, no current flows. Diodes are rated according to the amount of current they can handle without damage and the maximum reverse voltage they can sustain. A band on the diode usually marks the "-" end. The triangle on the diode's schematic points in the direction current is allowed to flow. TuteBot uses diodes to isolate parts of the circuit and short out induced voltages of the wrong polarity.

A single-pole, single-throw (SPST) switch is shown at the left of the second row in Figure 2.12. Switches are characterized both by the number of connections that can be made or broken by moving the switch lever and by the number of different lever positions that make contact. A single pole, single throw (SPST) switch is the simplest type of switch. With the switch lever in one position, connection between its two leads is broken. In the other position, connection is made. An SPST switch might serve as the power switch for TuteBot, if desired.

To detect collisions, TuteBot uses a *momentary contact switch.* These types of switches have an internal spring that endeavors to keep the switch in one state. As long as the switch lever or push button is pressed, the switch circuit is closed. When the lever is released, the circuit opens. Momentary contact switches with the opposite sense (open when pressed, closed when not pressed) are also available.

Resistors impede the flow of current. Their ability to do this is measured in ohms, Ω; kilohms, $K\Omega$; or megohms, $M\Omega$. The current, I, that will flow through a resistor with resistance, R, given an applied voltage, V, is $I = V/R$. This is known as Ohm's Law. When current flows through a resistor, it must dissipate power. A resistor's capacity for dissipating power is measured in watts. In general, a resistor with a higher wattage rating will be physically larger than one with a smaller wattage rating.

To block direct current but allow the passage of alternating current, one uses a *capacitor*. Once connected to a voltage source, such as a battery, current flows into the capacitor until it has accepted as much charge as it can. This ability to accept charge is usually measured in units of micro- or picofarads (μF or pF). If the voltage supply is removed from the capacitor, the stored charge keeps the voltage across the capacitor constant. Shorting the leads together causes a current to flow until the charge is depleted and the voltage across the capacitor goes to zero. TuteBot uses capacitors as memory cells. The presence or absence of stored charge represents the robot's recent history, or state.

There are many different capacitor technologies. Most capacitors can be connected into a circuit without regard for polarity. One type for which polarity is important is the electrolytic capacitor. The leads on these capacitors are marked "+" and "-" so that it is clear which way they should be inserted into the circuit. Electrolytic capacitors can generally store more charge in a smaller volume than other types of capacitors. The maximum voltage that can be applied to a correctly connected capacitor before damage occurs is listed as the WVDC (Working Voltage, Direct Current).

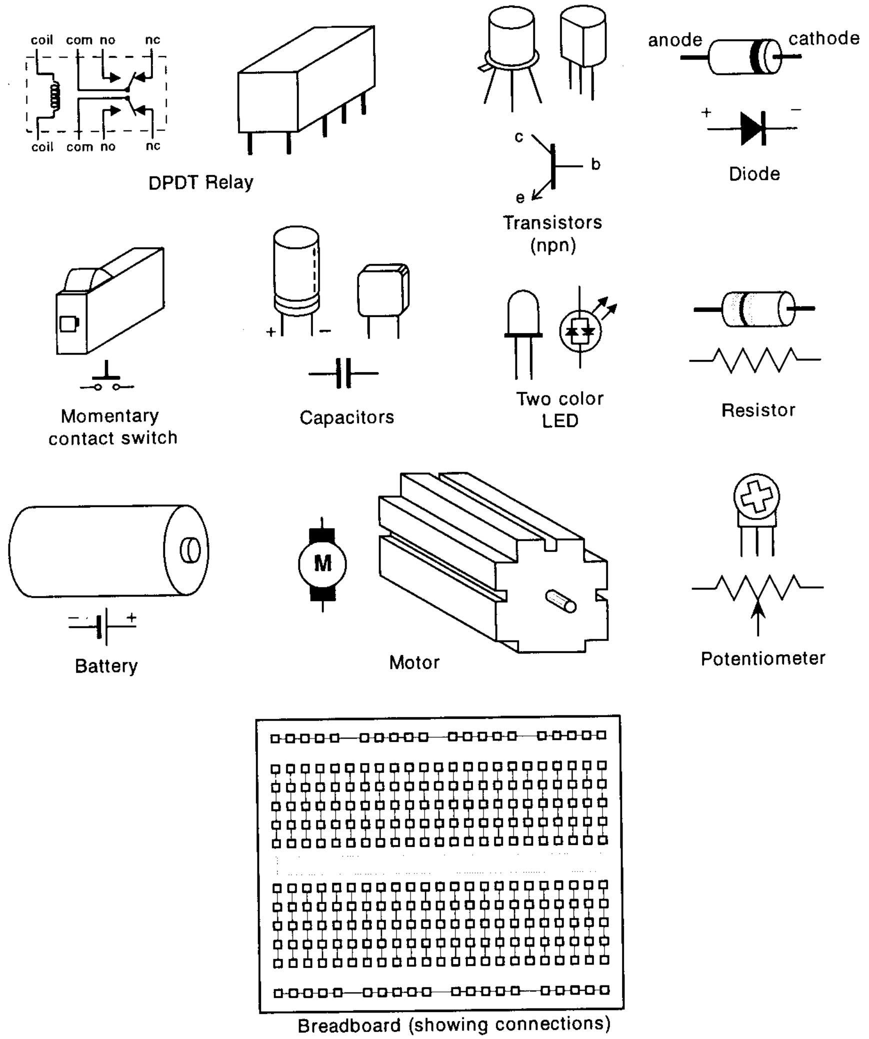


Figure 2.12. The relationships between schematic symbols and the physical components they represent. All of these components are used in TuteBot's brain. No other components are necessary, and the entire circuit will fit in a 6-inch-long breadboard mounted on top of TuteBot's chassis.

A *potentiometer* is simply a resistor whose resistance is adjustable. As with fixed resistors, there are a large number of resistances and maximum power ratings from which to choose. A potentiometer allows the user to manually alter some parameter of a circuit. We will use potentiometers in TuteBot to control its response to collisions—how long it backs up and how long it turns in place before proceeding forward again.

The first item found in the third row of Figure 2.12 is a *battery*. Batteries supply currents required at some characteristic voltage. The nominal voltage rating of a battery is normally stamped on its case. TuteBot, for instance, uses six 1.5 volt (V) alkaline batteries. Many toys and portable appliances use nickel cadmium (NiCd) batteries which come in 1.2 volt cells.

Motors convert electrical energy to mechanical energy. Fischer-Technik motors were chosen for TuteBot because they are easy to integrate into the chassis and they happen to provide sufficient power for this application.

The last component in Figure 2.12 is an electronic *breadboard*. Internal connections among its sockets are shown. A breadboard allows us to quickly connect components into a circuit and to make changes easily. Vertical columns are connected together, as are the top and bottom horizontal rows. Typically, one would connect these rows to power, the positive side of the battery pack in this case, and ground, the negative side of the battery pack. The space between the columns in the center is the correct width to accommodate standard integrated circuit chips. The relays are the same width as standard chips.

Later on in this book when we discuss Rug Warrior, we will introduce a number of other components such as power MOSFET transistors, crystals, operational amplifiers, photoresistors, light emitting diodes, logic gates, microprocessors, memories, etc.

2.5 Electronic Construction

With those device descriptions as background, now let us look at the circuit for TuteBot's brain. Figure 2.13 gives the schematic. A schematic illustrates the topology of how all the electronic components are connected into a circuit.

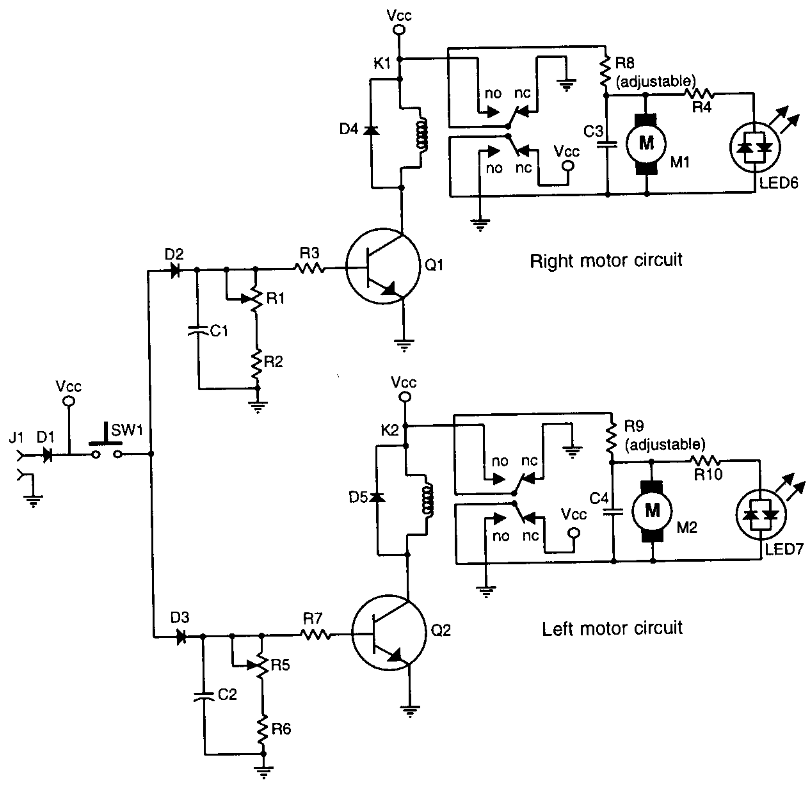


Figure 2.13. Schematic for TuteBot's brain.

In the circuit for a robot's brain, there are typically transducers connected on either side. For instance, on the input side, batteries and sensors act as input transducers. A battery converts chemical energy into electrical energy, and a sensor converts a physical phenomena from a mechanical form (say, the force acting on a bump switch) to an electrical form. On the output side, motors, speakers, lights, etc., act as output transducers. The motors on TuteBot convert electrical energy into mechanical energy. In between the input and the output transducers is the electrical circuit, which does the information processing. The time variation in the signals, the voltages and currents in the circuit, provides information transfer.

In describing a circuit's behavior, one usually speaks of voltage across a device and current through it. One bit of confusion can arise due to a verbal shorthand of speaking of such things as "the voltage at point A". What is meant and what would be more precise would be to speak of "the voltage across the network between points A and ground." The verbal shorthand comes about because ground is usually taken to be the reference, 0 volts.

The basic idea of TuteBot's circuit is that the front bumper switch (SW1 in Figure 2.13) generates a signal that tells the robot to back up. This bump signal is sent to each half of the circuit. The diodes D2 and D3 act to separate the circuit driving the left motor from the circuit driving the right motor so that they can independently have specifiable time constants for how long each wheel should back up. The time constants are implemented with resistor-capacitor (RC) circuits that hold a voltage for a given amount of time, depending on the values of the resistor and capacitor. The timing signals from these RC networks then direct the motors to reverse direction for the specified amount of time. Some driver circuitry to condition the signal to provide enough current to drive the motor has to be added at this point. This motor-driver circuitry is implemented with transistors and relays. A bank of resistors may be added in series with one motor to regulate its speed in comparison to the other motor.

There are two ways to proceed at this point. One is to go ahead and just build the circuit and not worry about understanding how it works. Simply build it, mount it on TuteBot's chassis, plug in the connectors and start playing with various behaviors by tweaking

Electronics:
1 Breadboard - Experimentor 350, Radio Shack 276-175
2 DPDT 5 volt relays - Omron G5V-2-H-DC5 (DigiKey Z770-ND)
2 Transistors - 2N2222A
2 2200 uF Capacitors, electrolytic, 10 WVDC, radial lead - DigiKey P6219-ND
[Physical size of cap is important, substitute must not
be bigger than above but may be smaller e.g. 1000 uF]
2 0.1 uF Capacitors, 50 WVDC, radial leads - Radio Shack 272-109
[Substitutions OK, but try to keep as small as possible]
2 1 K potentiometers, Bourns 3352T, DigiKey 3352T-102ND
[Must be finger adjustable]
5 Diodes, 1N914
2 33 Ohm Resistors, 1/2 watt
2 82 Ohm Resistors, 1/4 watt
2 270 Ohm Resistors, 1/4 watt
2 220 Ohm Resistors, 1/4 watt
2 Dual-color LEDs, Radio Shack 276-012
[Substitution may require adjusting the 220 Ohm resisitor]
3 12" lengths 22 AWG solid conductor wire: Red, Black, Yellow

Figure 2.14. Use these parts to construct TuteBot's brain. A good understanding of how the circuit functions will allow the builder to make substitutions. Radio Shack part numbers are given in parentheses. Where no part number is given, any component with the listed parameters can be used.

potentiometers and adding resistors in series with the motors. The other way is to convince yourself you understand every last detail of the circuit configuration before you start stripping wire.

We recommend a quick skimming of the circuit description and then directly putting the circuit together. The parts list for the circuit is given in Figure 2.14, and because the purpose of this chapter is designed to overcome the inertia of getting started, an exact layout on a Radio Shack breadboard is shown in Figure 2.15. Build the circuit just like this one and TuteBot should work. One can then go back through the circuit observing voltage signals across various portions of the network with an oscilloscope to compare traces to graphs for a better understanding.

A photograph of a finished breadboard is shown in Figure 2.16. One detail to note in assembling this circuit is that all the parts and pieces must make firm connection in the breadboard. Relays may be too short to make good contact when inserted into the breadboard and may need to be plugged into a dual inline pin (DIP) socket before being plugged into the breadboard. A 16 pin DIP socket ought to be sufficient. Use care when installing the diodes and the electrolytic

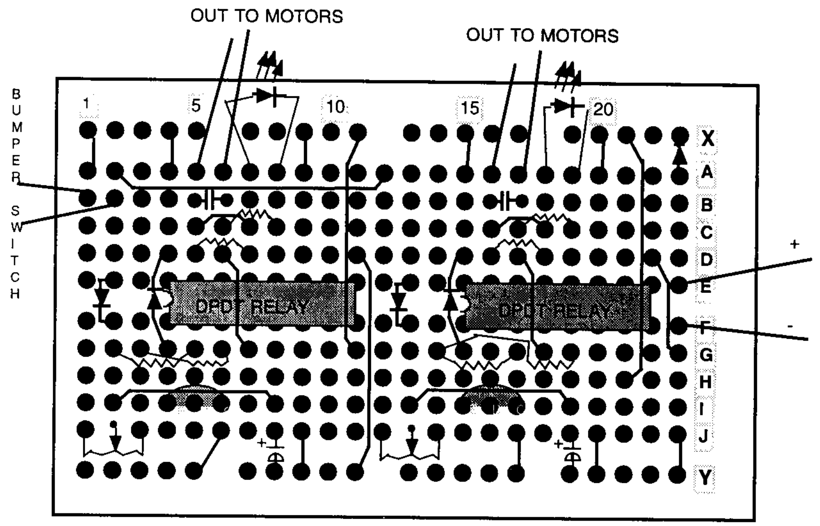


Figure 2.15. One possible layout of the TuteBot circuit.

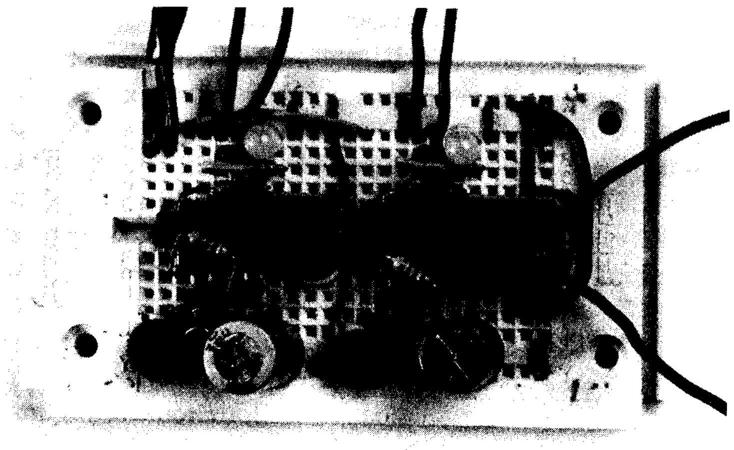


Figure 2.16. Details of the breadboard.

capacitors. These devices are polarized. If they are installed the wrong way they may be damaged.

It is a good idea to test the circuit as you go. Build only half of it first and check to see that it drives the motors as desired. With power applied and the motor not connected, check to see that pressing the bumper switch activates the relays. If operating properly, a click will be heard. The bias resistors, R3 and R7, may need to be adjusted if relays or transistors other than the ones specified are used. If the relay does not operate, choose smaller resistors until it does (but don't go below about 100 Ω.)

In general, it pays to be neat when breadboarding a circuit. Any time saved in quickly throwing together a sloppy circuit is usually more than wasted in debugging time. Cut and strip wires to appropriate lengths so they lie flat on the breadboard. Buy lots of different colors of hookup wire and stick to conventions for power and ground. If you use red for +9 volts and black for ground, then it becomes easy to visually check your breadboard, because all wires connected to the top horizontal row should be red and all the wires connected to the bottom horizontal row should be black.

Another important tip before connecting power is to always "ohm out" power and ground–that is, check with an ohmmeter that power and ground have not been inadvertently connected together on your breadboard. This prevents smoke from streaming out of your circuit. Never remove components with the power on. Power down first. If the circuit does not work, first check with a voltmeter that all points in the circuit that should be connected to power are actually at +9 volts, and that all points which should be at ground actually read 0 V. While this all sounds rather obvious, you would be surprised at how many problems are caught by these few simply steps.

2.6 Operation

For a more detailed exposition of the TuteBot circuit of Figure 2.12, we break the system into modules and explain each piece. The circuit is divided into two nearly identical halves. For simplicity, we describe only one half, the rear half which controls the right motor.

As soon as power is applied by plugging in the black (green) wire to the negative battery socket after having plugged in the red wire to

the positive battery socket, both motors begin to turn forward and TuteBot is able to move straight ahead. If we look at the portion of the schematic showing the right motor's connection to its relays, we see that there is a lever arm that can switch between normally open and normally closed connections. This type of relay is a double pole, double throw (DPDT) relay. It is the only electronic component not available from Radio Shack in a 9-volt variety. We can see for the right motor that the normally closed connection applies 9 volts across the motor. If the motor moves in the reverse direction, switch the leads going to the motor and it will then move forward. This is true for the left motor also. Notice the Light Emitting Diode (LED) is green when it is going forward and red when going in reverse.

Again, looking at the right motor-portion of the circuit, if TuteBot strikes an obstacle and the bumper switch is closed, a current flows through diode D2 charging capacitor C1. Simultaneously, current flows through resistor R3 into the base of transistor Q1. The base current causes Q1 to conduct—pulling current through the coil of the DPDT relay. When the current is provided to the relay, it switches from the normally closed state to the normally open state. The motor terminal previously connected to +9 volts is now connected to ground, and the other terminal which was previously connected to ground is now connected to +9 volts. This causes current to pass in the opposite direction through the motor, making it spin in reverse. The LED should be red while motors are in reverse.

As the reversing motors cause TuteBot to back up, its bumper is no longer pressed against the obstacle and the bumper switch, SW1, is no longer closed. With the switches open, the RC circuit is no longer connected to +9 volts. However capacitor C1 continues to supply current for a while to the base of the transistor and the motor continues its reverse rotation. The capacitor discharges at a rate controlled by the resistors. At some point, Q1 ceases conducting, the relay opens and the motor resumes its forward rotation. Diodes D2 and D3 isolate the circuits so that the capacitor can discharge at the desired rate (so that current cannot drain off C1 and begin charging the left motor's RC circuit).

Figure 2.17 illustrates how the voltage across the right motor's RC network changes with time. With the switch closed, the battery charges the RC circuit (this voltage is taken as between point A and ground) up to V_o. When TuteBot backs away from the obstacle and the switch is opened, the voltage across the capacitor falls at a rate determined by the values of the resistor and the capacitor. To be more precise, this relationship is $V = V_o e^{-t/RC}$, where V_o is the power supply voltage. Figure 2.17(b) illustrates the RC network connected to the left motor. The smaller resistance in (a) causes the current to drain away more quickly, keeping the robot's right wheel in reverse for a shorter time period than the left wheel. This causes the robot to turn to the left.

The right motor turns in reverse for a period of time, which is determined by the following factors:

- The size of capacitor, C1.
- the value of bias resistor, R3.
- The amplification factor of transistor, Q1.
- The resistance of the potentiometer, R1.
- The current level needed to activate relay 1.

A very brief motor reversal may be selected by setting the potentiometer to its smallest value. A reversal longer than the one available in the circuit is most easily achieved by increasing the value of C1, as it is actually the product of R and C which sets the time constant. You can see this time lag as the duration during which the two LEDs(6,7)are different colors (one red, the other green).

We can see how the changing currents set up by the RC network are able to activate and deactivate the transistor Q1 by referring to Figure 2.18. Depending on the characteristics of the particular transistors and associated circuit components, a transistor can be used as either an amplifying device or as a switch. The TuteBot circuit requires the transistor to act as a switch as shown in Figure 2.18(a). When base current is supplied, the switch closes and the load draws current because it is connected between power and ground (see Figure 2.18(b)). Our very simple model of how a transistor switch works shows that as long as the current flowing into the base of transistor Q1 is greater than or equal to $i_{b,sat}$, the switch is on and current

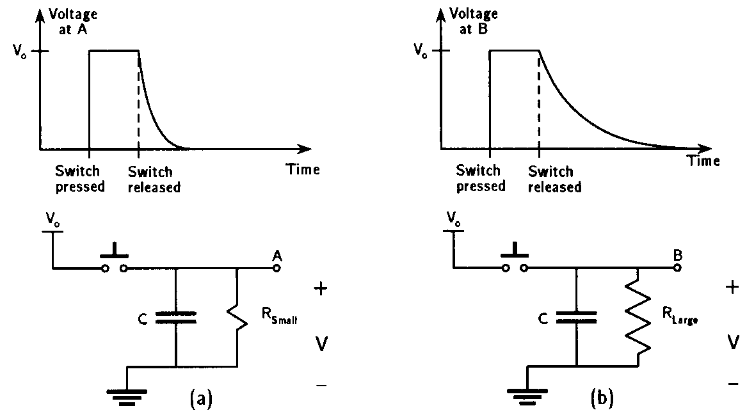


Figure 2.17. As long as the momentary contact switch is pressed, the voltage between point A and ground or point B and ground will be equal to V_o. When the switch is released, charge begins to drain from the capacitor through the resistor. The small resistance in (a) drains the capacitor more quickly than the large resistance in (b).

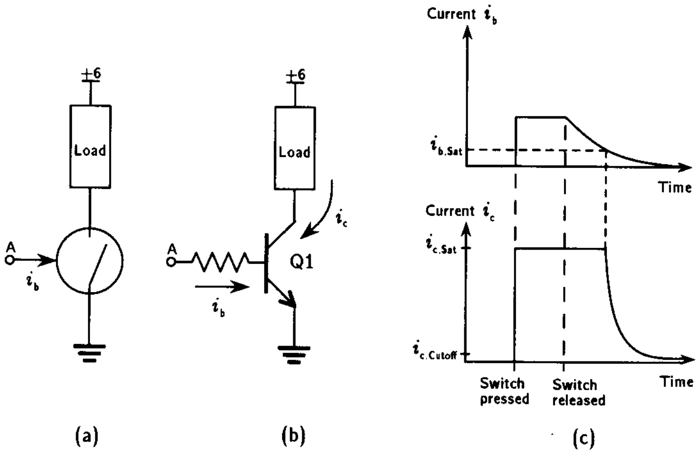


Figure 2.18. (a) A transistor is modeled as an ideal switch. (b) In reality the base current is set by the base resistor's value for a given voltage applied to terminal A. (c) The base current must be large enough to put the transistor into saturation (turning it fully on).

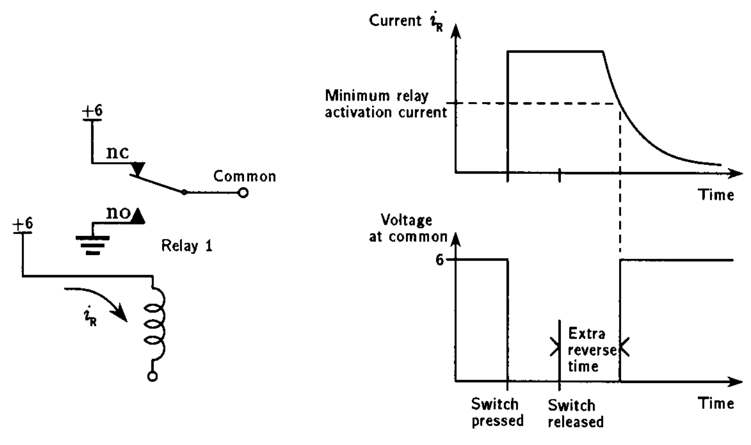
flows through the load. When the transistor's base current falls below $i_{b,sat}$, the transistor switches off and no current flows through the load. A small base current is able to control whether or not a large load current is allowed to flow. In Figure 2.18(b), we see that a base resistor is needed to set the base current for the transistor switch. The timing signals of the current flowing thorough the base resistor are shown in Figure 2.18(c). For the duration of time that TuteBot is in contact with the obstacle and the RC circuit is charged up to V_o, the base current is large enough that the transistor is completely on and saturated—that is, the collector current has reached its maximum possible level, $i_{c,sat}$.

As TuteBot backs up from the obstacle, the bumper switch opens and the voltage drains off the RC network, the current through the base becomes smaller. Eventually, it falls to $i_{b,sat}$, where the transistor begins to come out of saturation. The collector current falls to 0 and the load becomes open circuited. Actually, a small amount of current does continue to flow for a while even when the transistor is "off". The transition from "on" to "off" is not quite as sharp as in a real switch.

When the transistor switches on, it draws current, i, through the coil of the relay as shown in Figure 2.19(a). Current through the coil creates a magnetic field which forces the relay lever to move. The relay lever then switches the common connection (attached to one terminal of the motor) from normally closed to the normally open pin. This happens in the relay associated with each motor, reversing the polarity of the voltages applied across them. For all the time that Q1 is on, current is pulled through the relay causing the motor to switch from forward motion to reverse motion. The LED subsequently switches from green to red.

The essential difference between the left and right motors is the relative times at which they turn off their reversing behaviors. In Figure 2.19(b), we can see the timing diagrams of the current through the relay and the resulting voltage applied between one motor terminal and ground.

First, as the transistor Q1 turns off, it causes load current to stop flowing. This takes some amount of time after the bump switch is released due to the time delay set up by the RC circuit. When the current through the relay falls to a level which can no longer



(a) (b)

Figure 2.19. (a) The amount of current, i_R, flowing through the coil of the relay determines whether its common terminal is connected to its nc, normally closed, or its no, normally open terminal. When i_R falls below minimum activation current the state of the relay changes. (b) The "Extra reverse time" is the extra amount of time the motors run in reverse after the bumper switch has been released.

sustain the necessary magnetic field to keep the lever attracted to the normally open pin, the relay switches back to its normally closed configuration. This occurs to the relay attached to the right motor.

The lower graph in Figure 2.19 (b) shows the resulting voltage change over time for one of the right motor's terminals. The other motor terminal, normally at 0 volts, switches to +9 volts when the bump switch hits an obstacle and reverts to 0 volts again (after the time lag set up by the RC network) after the bumper is released.

A similar mechanism is implemented on the left motor except that its potentiometer, R5, is tuned to give a different time delay for the right motor. The robot can thus be programmed to turn more or less sharply by adjusting the potentiometer settings for each wheel.

Four other points are worth mentioning concerning the right motor circuit of Figure 2.12. The first is the appearance of diode D4 across the DPDT relay. The reason for adding this device to the relay is that the diode protects the circuit from the large voltages that are induced by collapsing magnetic fields in the relay coils when the transistor turns off. If diode D4 were not there, the inductance of the coil would try to force the current flowing through it to keep flowing down through transistor Q1. Because Q1 has been opened, current through the coil results in an increase in voltage at the collector of Q1. If this voltage exceeds the maximum rating that the cutoff transistor can withstand, it becomes damaged or blows up. The diode alleviates this problem by providing a return path for the coil current when the transistor turns off.

The second point to note in the final circuit is that the capacitor C3 has been placed across the terminals of the motor. This capacitor attenuates the voltage spikes produced by the motor. Typically, these capacitors are soldered directly to the motor terminals rather than placed back at the circuit board.

Third, note that the directional LED (Light Emitting Diode) turns green when the TuteBot is going forward and turns red when the TuteBot is going backward. Likewise note that when TuteBot is turning, one LED is red and the other green. The LED is parallel to the motor in the TuteBot circuit.

Finally, note that a "resistor bank" could be connected in series between the relay and the right motor. This bank is for matching the speeds between the two motors. Which motor should be connected to the resistor bank is something which must be determined by experiment. Although the motors and geartrains are supposed to be identical, in reality they are not.

These differences manifest themselves as mismatches in the speed at which the wheels turn. To make the adjustment, power-up the TuteBot on and allow it to roll across the floor. It will make a long arc in one direction or the other. If it turns to the left, then the right motor is turning faster; attach the right motor to the resistor bank in series. Otherwise, attach the left motor. With n resistors wired in parallel, the total resistance R_T, of the resistor bank increases as each resistor, R, is removed: $R_T = (1/n)R$. The more resistance we place in series with the motor, the less current will flow and the slower the motor will turn. Add or remove resistors until both motors rotate at the same speed.

TuteBot is now complete and ready to go. Try running it in a few different environments. Try adding the wall-following behavior discussed earlier to bias the motor speeds so the TuteBot travels

forward in an arc by inserting a resistor bank. If TuteBot goes too fast and falls apart when it crashes into things, electric tape, double sticky tape, Velcro^{TM}, and glue work wonders with breadboards and with Fischer-Technik components.

Have fun!

2.7 Exercises for the Reader

With the wall-following behavior implemented as described above, the robot will simply turn in circles if it is set in motion far from a wall. As an exercise, try to devise an additional behavior (possibly requiring another component or two) which will cause the robot to go straight until it encounters a wall and then begins to follow the wall.

Think about all the different ways you might add one or more photoresistors (response to light) to the TuteBot circuit. How about a thermistor (response to temperature)? What behaviors are produced in each case? Can you make a TuteBot that follows a light such as a flashlight?

2.8 References

While the TuteBot exercises in this chapter were designed to be simple examples to get started, it might be the case that many people feel more at home with a computer-controlled robot than with the analog electronics of TuteBot. If so, proceed to the next chapter describing Rug Warrior's microcontroller brain. However for background in electronics, the bibles for robot builders are Horowitz and Hill (1989) and the associated student manual (Hayes and Horowitz 1989), which give extensive practical information on analog electronics in very readable presentations. *The ARRL Handbook for the Radio Amateur* (Kleinschmidt 1990) is another very good source for the beginner new to electronics. For articles and reports on simple robots and how to build things, a few pieces have trickled out of the MIT Mobile Robot Lab over the years. Jonathan Connell (1988) describes Photovore, shown in Figure 2.20, a light-eating, dark-avoiding, relay-driven robot using three photoresistors and a Radio Shack toy car

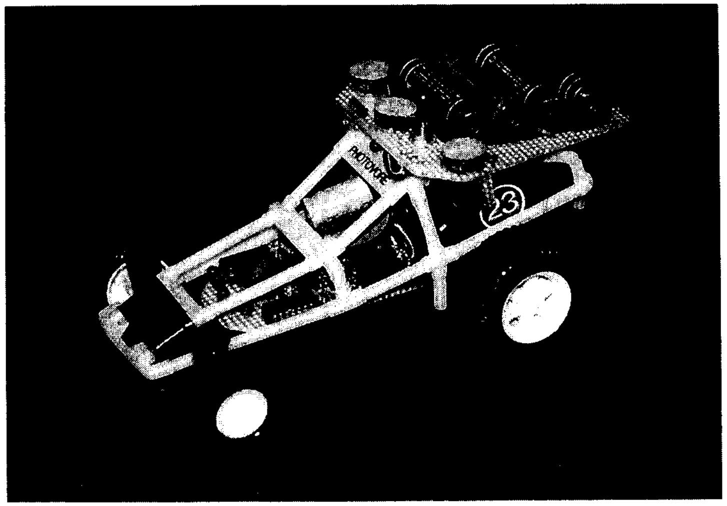


Figure 2.20. This MIT robot, known as Photovore, performs an interesting set of light seeking behaviors. It uses only analog circuitry to achieve its behaviors.

base. Photovore is also described in *The Olympic Robot Building Manual*, (Flynn et al 1988), from which this book grew. A picture book of the resulting talent show robots is contained in Flynn (1989). Another minimalist mobile robot is described in the August 1991 issue of *Popular Electronics*, (Connell 1991). Kits and printed circuit boards for building your own version of Photovore can be purchased from Johuco, Ltd. See Appendix C for addresses and phone numbers in the list of manufacturers. Also, please note that kits are available from A K Peters, Ltd. for TuteBot and Rug Warrior.

3

Computational Hardware

The elementary circuit that controls TuteBot serves its purpose well. Using only relays, potentiometers, bump switches, and some discrete components, TuteBot is able to avoid obstacles and follow walls. Adding a few more sensors and continuing in the same vein of using hard-wired logic for the intelligence system, many other interesting behaviors can also be designed. Rather than pursue this route, however, we now introduce a more sophisticated control element, the *microprocessor*. It has a number of advantages over hard-wired logic in terms of versatility, power consumption, size, and ease of use.

Most importantly, however, the microprocessor introduces a significant new tool in solving the robot control problem: software. To change the behavior of robots of TuteBot's nature, we must adjust potentiometers, rewire circuits, and add or alter components. The behavior of a software-based robot, in contrast, can be changed by typing at a keyboard.

Hardware determines a robot's ultimate potential, but realizing that potential is the job of software. There is an intimate relationship between these two elements which we will try to make clear as we proceed. Organizing the software in the proper way is also important for simulating intelligent behaviors. The low-level interface between