ROBOTICS

MARK J. ROBILLARD,
ROBOTICS EDITOR

Experimental robot vision

LAST TIME WE PRESENTED THE CONstruction details of a simple photocell-based vision sensor. To conclude the presentation, this time we'll show you how to connect it to a microcomputer, and then we'll discuss a few software algorithms that demonstrate how to use it.

The sensor consists of nine photocells. For a computer to be able to read the analog resistance of each photocell, that resistance must be converted into the digital language the computer understands. So we must use an ADC (Analog-to-*Digital* Converter). When a photocell is connected between the positive supply voltage and a resistor that is grounded on the opposite end, the pair acts as a variable voltage divider. The divider is variable because the resistance of the photocell changes depending on the amount of light that reaches its active surface. When you connect the output of the divider circuit to the input of an ADC, a digital representation of the voltage dropped by the photocell may be read.

It would be inconvenient and expensive to connect a single ADC to each of the nine sensors in our vision unit. Fortunately, however, National Semiconductor has an IC (the ADC0816) that includes not only an ADC, but also a 16-channel analog multiplexer that allows us to monitor all nine photocells (and seven other analog inputs, if desired) without multiplying costs nine times.

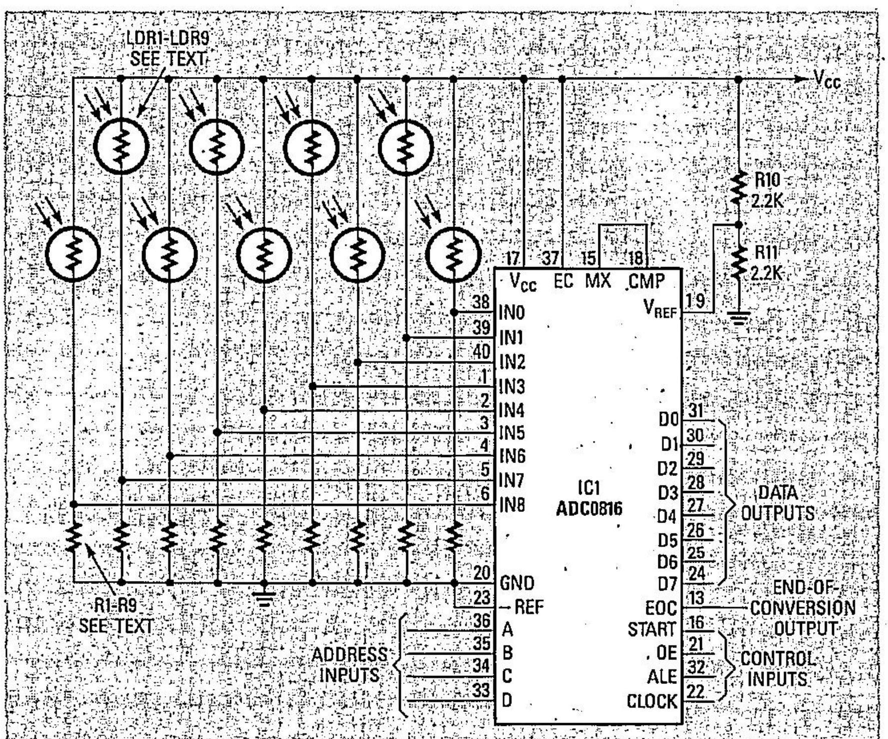


FIG. 1

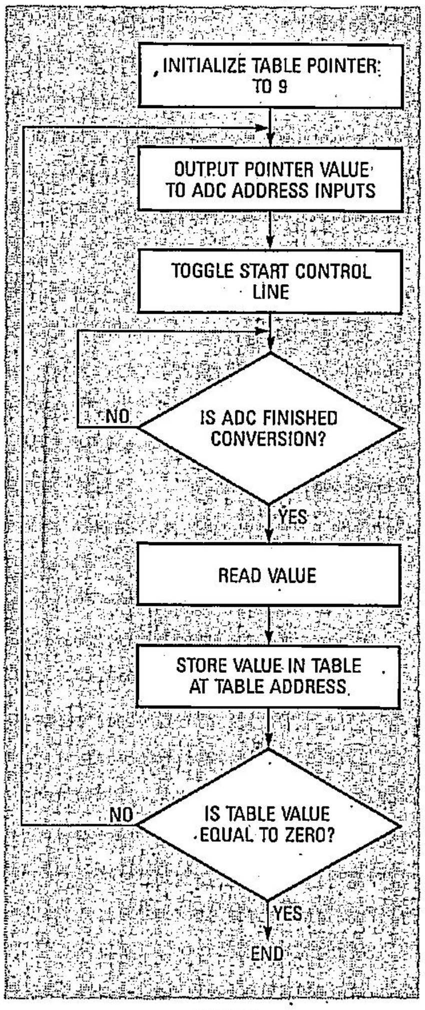


FIG. 2

Figure 1 shows the circuit details of the sensor interface. In addition to the sixteen analog inputs, the ADC0816 has four address inputs that allow you to select which of the sixteen inputs you want to read. Further, the circuit has several control lines that are used to select various operations. We'll discuss each of the control lines below.

If your computer has a built-in eight-bit parallel interface, you can probably use the circuit directly as shown. Otherwise, you'll have to add some external circuitry. One way of connecting the ADC0816 to an 8-bit computer system is as follows.

The computer's data bus (or eight-bit I/O bus) is connected directly to the IC's 3-state data outputs (pins 24–31). You could AND the computer's READ signal with a decoded port address and apply that signal to the OUTPUT ENABLE input (pin 21) in order to read a value from the ADC0816.

Selecting a channel is done by setting up the four address lines of the ADC0816 and then strobing the address into the ALE input (pin 32) via another decoded output AND-ed with the computer's WRITE line. The ADC's address lines can be connected directly to the low-order address lines of the control computer.

Last, the START input (pin 16) is used to start the conversion process. You could drive that input with another decoded output port AND-ed with the computer's WRITE line.

AN ENDING, A BEGINNING

This marks the 19th and final installment of Mr. Robillard's Robotics column. We'll miss Mark (who will contribute an occasional feature article), but we're happy to announce a new series of articles that includes complete details for building, operating, and experimenting with a personal robot. The new series begins in this issue and will continue for many months, as we continue keeping you up to date on the latest developments in the fascinating field of personal robotics. **R-E**

Because the ADC works much slower than your computer, you cannot simply select a channel, send a "start" command, and then read the data. The ADC must sample the input and then convert it to digital form. The ADC0816 can take as long as 116 µs to complete the conversion. To alert the computer when the conversion is done, the IC has a special END-OF-CONVERSION output (pin 13) that goes high when a digital representation of the analog input may be read. You can monitor pin 13 by AND-ing a decoded I/O port address with your computer's READ line. Alternatively, you may want to connect pin 13 to an interrupt input; doing so would allow your computer to do other things while the ADC is working.

Figure 2 outlines the basic algorithm for scanning the nine-element sensor. First we select analog channel one. Then the START signal is activated. Then the computer goes into a loop and monitors the END-OF-CONVERSION output. When that signal goes high, the output buffer is read, and its value is stored in a nine-byte table for analysis later. The program loops to select the next channel (i.e., the next sensor element) and executes the same sequence of operations. When all sensors have been read, the algorithm is finished.

After reading in the data, it must be analyzed. It would simplify analysis if each sensor returned a value of 1 for light areas and a value of 0 for dark areas. Then the table of bytes could be compared with a set of previously-stored templates. The program would interpret the object as being the one with the closest match.

However, things don't work quite so simply in the real world. In fact, the circuit shown here is so sensitive that, instead of getting two distinct values that represent light and dark, you'll be getting readings with 256 distinct values. Areas of your target that appear to be the same will actually register tremendous differences.

In order to eliminate most of that "clutter," an auto-sensitization adjustment must be made. What we must do is to trick the circuit into being less sensitive. One way of doing so is with a threshold adjustment. Looking back at the circuit in Fig. 1, notice the resistive voltage divider connected to pin 19 (V_{REF}) of the ADC. All converted values are compared to the value at that pin. By varying the reference, you can adjust the sensitivity of the circuit.

You could use a DAC (*D*igital-to-*A*nalog *C*onverter) to perform the auto-focus. Connect the digital inputs of the DAC to a separate output port, and the analog output of the DAC to the reference input of the ADC. Then, by placing a black and white cross (or some other shape) under the sensor, have the computer read the ADC. If the sensors under the black areas don't read similarly, have the computer change the reference voltage via the DAC converter. Adjust the reference until the output reads the way you want it to.

Also, you could calibrate the sensor manually using a potentiometer and some sort of program that outputs the values to the screen. The value of the automatic circuit, though, is that the computer can calibrate itself at any time.

I have found the sensor to be great at picking out brightly colored symbols on a dark block. In addition, the sensor reads well at a distance of one inch above the target object. As discussed last time, use a flash from an old camera to illuminate the area evenly. And be sure that the duration of the flash is at least 200 µs to compensate for the conversion time. **R-E**

ROBOTICS

Ultrasonic vision

MARK J. ROBILLARD
ROBOTICS EDITOR

FOR A ROBOT TO EXHIBIT EVEN A modicum of intelligence, it needs some type of feedback mechanism between the "thinking" part and the "doing" part. Without some type of feedback, the "thinker" can't adjust the actions of the "doer" in response to changes in its environment.

The most useful form of feedback for human beings is visual. Machine vision, however, is limited by current technology in its ability to resolve details and in its ability to estimate distances.

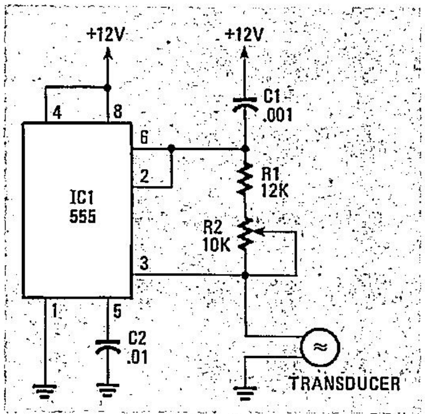


FIG. 1

There are several dozen companies now selling machine-vision systems to industrial manufacturers. Those systems are generally used to inspect finished goods. There are a few companies selling devices that will help a mobile robot get around in a cluttered room. With that type of vision, a robot can see what it is gripping, where it is going and where it has been.

Human beings have a crude, built-in distance measuring system that makes use of the eyes as well as stored knowledge of how

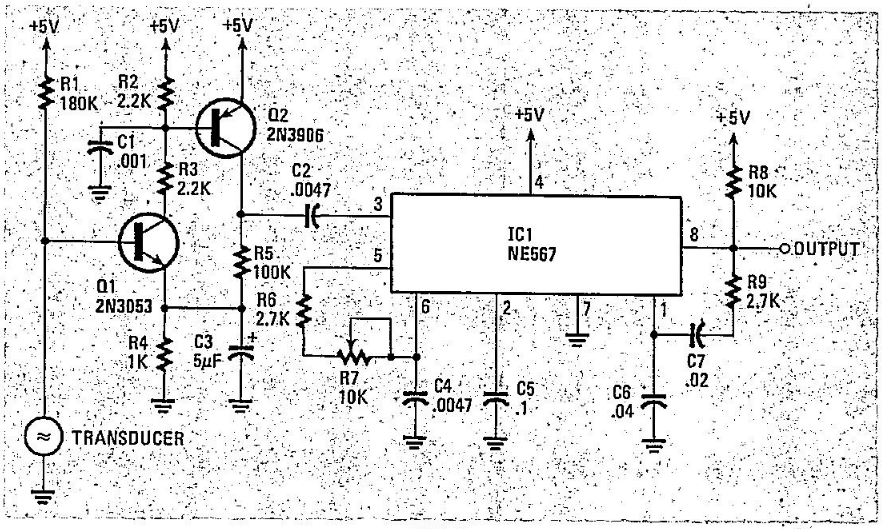


FIG. 2

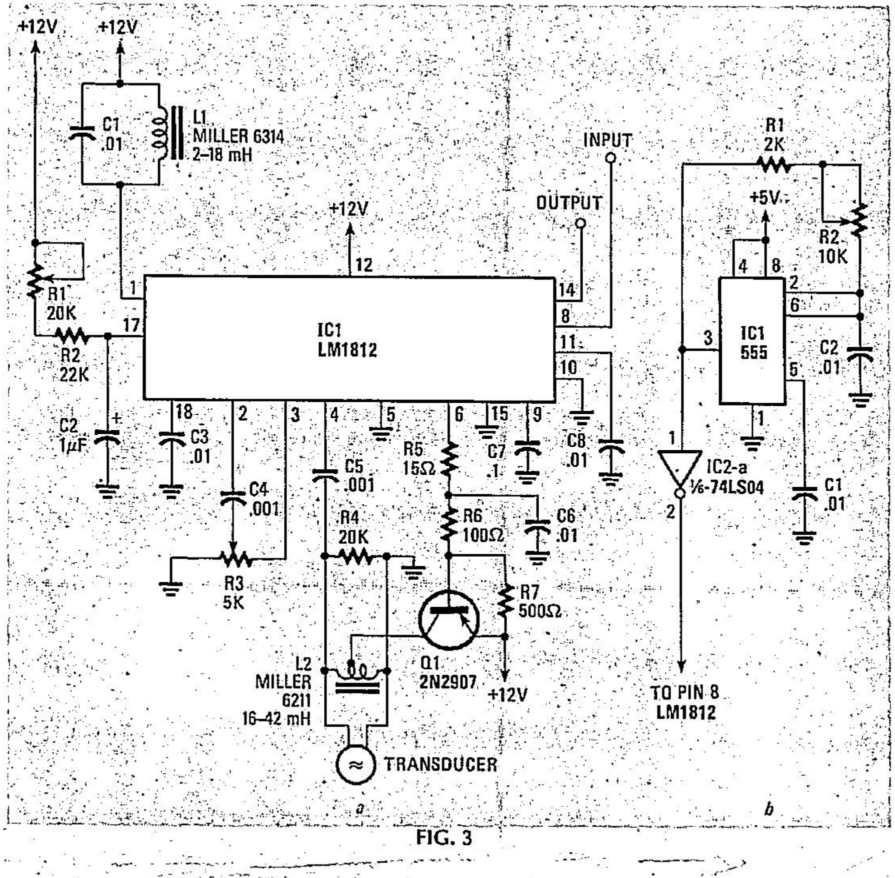


FIG. 3

an object looks at different distances. We then compare the known image to the one we are presently encountering and *voilà*—we obtain a rough estimate of the object's distance. That is the method used by systems that inspect parts on a finished-goods assembly line. We could apply that method to a robot system, but that would require auto-focusing lenses and complex image-analysis algorithms in the controlling electronics. However, there is a simpler and more economical way to obtain distance information: *ultrasonic ranging*. That method is frequently more accurate than light-wave systems.

Ultrasonics theory

In a typical ultrasonic ranging application, an ultrasonic transmitter starts a timer as it emits a short burst of 40-kHz sound waves through its transducer. Those sound waves spread out as they travel through free air. When they strike an object in their path, some of the sound is deflected back toward the transducer, which now acts as a receiving device, like a conventional audio microphone. The received sound waves are sent to a discriminator circuit that stops the timer.

The value in the timer is proportional to the length of time it took the sound waves to travel to the object and return to the origin. We can determine the distance (D) from the period measured by the timer (T) by using the classic formula, D = V/T. V here stands for velocity, and is simply the velocity of sound waves in air, about 13,080 inches per second. Basically that's all that's needed to measure distance. So let's look at some practical implementations of both transmitters and receivers.

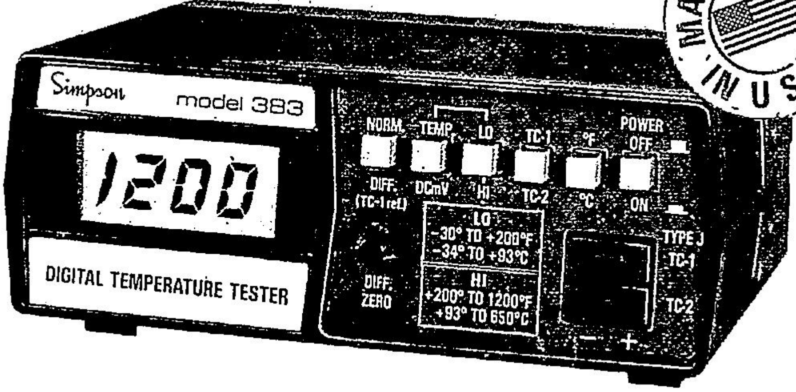
Ultrasonic circuits

A simple 40-kHz transmitter is shown in Fig. 1. The 555 is a very inexpensive eight-pin timer IC configured as an astable multivibrator oscillating at a frequency of 40 kHz. (That frequency can be adjusted by potentiometer R2.) With a twelve-volt supply, the 555's output will swing about ten volts, which should be enough to drive common transducers. Any 40-kHz oscillator could be substituted for the circuit shown in Fig. 1. In fact, it would be quite simple to generate a 40-kHz signal with a computer, and drive a transducer with that signal through an open-collector TTL gate (such as a 7406).

The receiver portion of the system is only a little more complex. In the circuit shown in Fig. 2, the received signal is amplified by a discrete transistor amplifier, and then applied to a tone-decoder NE567. That IC, a phase-locked-loop contains its own internal oscillator that must be adjusted to match the transmit frequency. The NE567 has a comparator that will send the output high when it detects an incoming signal equal in frequency to its internal oscillator.

The circuit shown in Fig. 3-*a* uses a single IC specially developed by National for use as an ultrasonic transceiver. Using such a device allows you to use a single transducer for both transmit and re-

continued on page 108



ROBOTICS

continued from page 90

ceive functions. In addition, the same tuned circuit is used for both transmitting and receiving, so you don't have to worry about frequency mis-matching due to component drift.

Transmission is initiated by applying a logic "1" to pin 8 of the LM1812. The receiver will be disabled, and pin 14 will be low. To receive, pin 8 is simply grounded. Pin 14 will go high after the circuit receives an echo of the signal.

The time between initiating transmission and pin 14's going high is then equal to the time required to travel from the transmitter to the object and back. So to obtain the distance to the object, that time would be halved and plugged into the formula D=V/T (V=13,080). The circuit shown in Fig. 3-*a* has a range from 4 inches to about 6 feet. By adding the 555 circuit shown in Fig. 3-*b*, the low end of the range may be decreased to 3 inches, and the high end increased to 20 feet. Potentiometer R2 should again be adjusted to a frequency of 40 kHz.

That circuit is so simple and so inexpensive that you could build several for one robot.

Polaroid's ultrasonics

Polaroid Corporation has been using ultrasonic ranging in cameras such as the *SX-70* for some time now. Distance information obtained through ultrasonics is used to focus the camera's main lens automatically. The transducer used has a larger diameter and is slimmer than the type usually found in electronics surplus stores. Several years ago Polaroid packaged the electronics from their cameras along with a good tutorial manual, as well as an interesting experimenter's board. The package is called the "Polaroid Ultrasonic Ranging System Designer's Kit," and is available for $165 from the Polaroid Ultrasonics Components Group (Polaroid Corp., 119 Windsor Street, Cambridge, MA 02139). It comes with a special interface that displays distance information on an LED readout. **R-E**

ROBOTICS

MARK J. ROBILLARD
ROBOTICS EDITOR

Robot brains

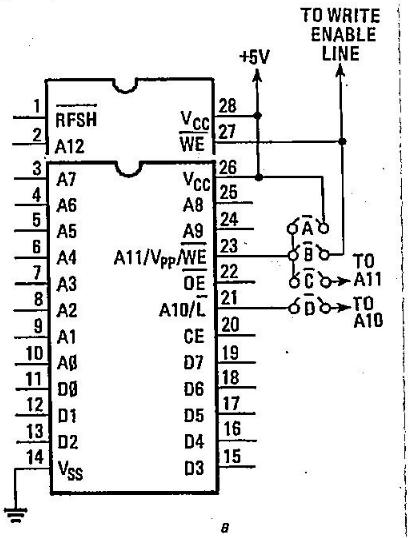
WE'VE TALKED ABOUT QUITE A FEW things the past few months, including putting together a robotics lab, robot motion and navigation, and last month we discussed, albeit briefly, the subject of voice recognition. It's time now to start putting those elements together—we need to give our robot some brains.

Perhaps you're asking, "Do I really need a computer to control my robot?" You could build a circuit out of discrete logic components that would allow your robot to perform the task you had in mind. But, as Hamlet said, that's the rub. To change that task, you would have to re-design and re-solder. However, if you implemented the control logic in software, you could change your design simply by keying in new instructions. Given the advantages of a software approach, there are still quite a few hardware questions to be answered. So what would be the composition of a suitable robot-control computer?

Microprocessors

The heart of any home computer is its microprocessor, and there are a number of different microprocessors to choose from. We must pick ours carefully, according to several criteria. The first is power. All devices used in robot-control circuits should be low power, assuming that the robot is to carry its power source (batteries) with it.

There are two chief methods of fabricating microprocessors: NMOS and CMOS. NMOS is the elder of the two, and NMOS almost invariably draws more power than the newer CMOS types. For example, an NMOS Z80 can draw as much as 150 mA of current at five volts. Most microprocessors (8080, Z80, 6800, 6502) were originally built using NMOS technology, although CMOS versions for many of them were introduced later.



DEVICE	TYPE	JUMPERS A	B	C	D
37000	ROM	X			X
37000	ROM			X	X
4118	RAM		X		
4802	RAM		X		X
4816	RAM				X
4864	RAM			X	X
2716	EPROM	X	X		X
2764	EPROM			X	X

b

FIG. 1

Bearing in mind the fact that the microprocessor must be supported by a bevy of other power-hungry components, before you know it, your computer's brain could easily require 1.5 amps of power. A small motorcycle battery would be necessary to keep a system with that sort of current drain operational for an hour! And that doesn't include power for the stepper motors!

It's easy to see why the industry almost universally uses CMOS microprocessors for portable computers. Not all microprocessors are available in CMOS yet, but the Z80, 6809, 8085A, 6502 and 8086/8 types are, and Motorola's 68000 will be soon.

The microprocessors mentioned above lead us to our next design criteria: eight or sixteen bits? (And 32 bits will soon be added to the "equation.") There is no clear-cut choice here. I have seen several robots designed with 16-bit microprocessors that should have been designed with 8-bit devices. Some people automatically assume that a robot requires the larger units, but that is not always the case.

Another consideration is whether the application demands a full-blown multi-IC design. In many cases a single-chip microprocessor will do the job. Single-chip micros typically have 128 bytes of temporary memory and 16 input/output lines. I find single-chip micros most useful as dedicated sensor controllers. For example, you could use one to control your robot's motors and sensors, while the main processor carried out the heavy-duty control logic. Now let's look at the memory question in a bit more detail.

Memory

Once again your application will determine how much RAM (*Random Access Memory*), ROM (*Read*

Only Memory) and EPROM (*E*rasable *P*rogrammable *R*ead *O*nly *M*emory) you'll need to provide. RAM is used for storage of temporary data, and data picked up by sensors; ROM and EPROM are used to store the control program and tables of data.

If you're new to this field, and if you're planning to build a research robot that will not be obsolete before you get done building it, then I suggest designing a "universal" memory system. That type of system is possible because there are, nowadays, RAM's, ROM's, and EPROM's that have almost identical pin-outs.

The circuit depicted in Fig. 1-*a* shows how a single 28-pin socket can be used to house several different types of memory IC's, and several different devices of each type. The table in Fig. 1-*b* shows how points *A, B, C,* and *D* in the circuit should be jumpered for each type of memory device. If you're interested in learning more about that subject, drop me a line. I'll send you a reprint of an applications note that discusses the subject in depth.

Ins and outs

Robot I/O (*I*nput/*O*utput) is, by far, the most involved decision. Depending on the sensors and motors in your system, I/O circuits can become quite complex. It will simplify matters if you use standard I/O schemes (like RS-232C) for communicating with peripherals or dedicated sensor controllers, like those mentioned above. There are many products on the market that communicate via RS-232C lines, and it's easy to interface your own devices via RS-232C.

Buy or build?

Whether you should buy a pre-assembled control computer or build your own also depends on the application. But if you think there's a chance that a pre-assembled unit will do the job, you'll save yourself a lot of hair-pulling by buying, rather than building.

Based on what I've said above, the specifications of an ideal robot-control computer might include a CMOS microprocessor, plenty of memory, standard I/O channels, and battery operation. Suppose I told you that you could buy an off-the-shelf device with a CMOS 8085A, 8K of battery-backed-up RAM (which is expandable to 32K), a real-time clock/calendar, an RS-232C port, a Centronics parallel port, a bar-code reader input, a full 60-key alphanumeric keyboard, and a forty-character by eight-line LCD display, all packaged in a case that measures about 8 × 11 inches?

That machine also has, built-in, a special version of Microsoft BASIC that supports all the I/O devices—and interrupts! You can burn your BASIC program into an EPROM and plug it into the ROM expansion socket that is provided. If you have an exotic I/O in mind, the entire system bus is also available.

The complete computer costs less than $350.00. If you haven't guessed what it is by now, you'll have to wait till next month. I'll have several interesting surprises then for those who choose to use that "mystery" computer for robot development. **R-E**

ROBOTICS

Tactile sensing

MARK J. ROBILLARD,
ROBOTICS EDITOR

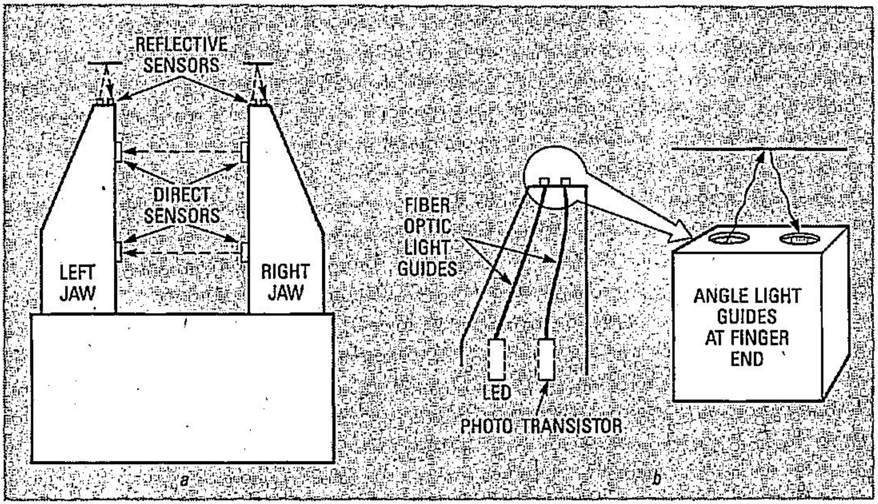


FIG. 1

MAN IS BLESSED WITH FIVE SENSES, AND one really proves its worth on a dark, cold winter morning. You roll out of bed and grope around for your slippers and bathrobe. Then you stumble into the kitchen, feel around for the light switch, and only then can you see what your hands and feet have been telling you. Oh that sense of touch!

Touch is important in robotics too. Most robots must move around to accomplish their tasks. Hence they need directions about where to go under different conditions. So they need to be able to acquire information about their surroundings. That information is obtained through a myriad of sensors—light sensors, sound sensors, and touch sensors, among others.

Different kinds of tactile information are important for different purposes. For example, often a robot must know how much force its gripper should apply to grasp an object. If too much force is applied, an object may be crushed. If too little force is applied, the gripper may not be able to maintain contact when it tries to lift the object. In addition, bump sensors around the perimeter of the robot's base can be used to detect collisions. The underlying technology of both kinds of tactile sensing can be the same. Let's see how.

Limit sensing

The simplest touch sensor is the common microswitch. A microswitch is often activated by either a long flat lever or a lever with an attached roller. Pressing the lever "makes" the connection. Actuation force is relatively small, so it takes little force for the robot to "feel" whatever the switch is contacting.

A robot gripper might be equipped with two microswitches, one in each jaw. With no object between the jaws, both switches would be open, but when an object is gripped, one or both switches will close. A computer interface could monitor the signals generated by those switches and it could use the various voltage levels to make decisions regarding the next action the robot should take.

Terence Thomas of Venice, FL sent a schematic diagram of a circuit that uses three microswitches. Two are used to detect obstacles around the perimeter of the body; a third indicates whether contact with the floor is being maintained. A floor sensor could be important to prevent a robot from making a perilous journey down a flight of stairs. Of course, the location of the floor sensor on the robot's body is critical. Two side-mounted microswitches provide direction control; when a switch on one side is activated, the robot turns in the opposite direction.

Optical sensors

Some sensing functions accomplished by switches can be accomplished better by another type of sensor. For example, rather than try to mount bulky, massive switches in the jaws of a gripper, you might use LED/phototransistor pairs, as shown in Fig. 1-*a*.

If you've ever been bowling, that setup should look familiar. The foul line sensor works on the same principle. It consists of a visible (or infrared) beam of light and a photo-detector. When your body breaks the beam, a light or a buzzer comes on.

Note that the sensors illustrated in Fig. 1-*a* are used in two different ways. The sensor pairs mounted between the jaws work like the bowling alley's foul-line sensor: an alarm condition is raised when the beam is interrupted. The sensor pairs mounted on the front edges of the jaws work on the opposite principle: an alarm condition is raised when the beam from the LED makes contact with the phototransistor.

As shown in Fig. 1-*b*, each member of the reflective pairs should be angled slightly inward so that a nearby object will reflect light from the LED back to the phototransistor. The phototransistor would be connected to a computer interface or other controller.

The reflective pairs can provide more than just an object-is-present/object-is-not-present signal. The amount of light that is reflected provides an indication of how close the object is to the end of the gripper. So the output of the reflective sensors could be processed by an A/D converter and fed to a controlling device which would then be able to locate objects. A control sequence might go as follows.

- Move the gripper to the approximate position of the target object.
- Scan the gripper from left to right until the object is detected by the reflective sensor at the tip of the right jaw. The gripper must be open so far.
- Move the gripper right until the object is detected at the tip of the left jaw.
- Move the gripper left again until neither jaw detects the object. The object is centered left-to-right now.
- Move the arm forward until the first inside-the-jaw sensor detects the object.
- Move the arm forward until the second inside-the-jaw sensor detects the object.
- Retract the arm until neither sensor detects the object, or until both detect the object if it is large. The object is centered front-to-back now.
- Close the gripper.

There is one flaw in that sequence of gripper motions. As the last step, the robot is to close the gripper to capture the object. But how does the robot know when to stop closing? In some cases, the maximum gripping force of a lightweight arm can be applied to the object. But that's not always the case. And how does the robot know when it has a good grip?

A force sensor

There are several ways to build a force sensor. One simple way is to use a spring and a microswitch. The spring is attached to a cable that winds around the shaft of a drive gear as motion occurs. The cable goes past the lever of a microswitch. When tension in the cable exceeds the actuation force of the microswitch, the switch contacts close. The device controlling the robot must then act on that signal. The spring in that sort of sensor must be selected carefully, according to the amount of force required to actuate the switch.

Other approaches use optical feedback. For example, the patented approach of Heath's *Hero 2000* uses two optical encoder disks. One is attached directly to the gripper, and the other is fastened by a spring to the first. As long as there is no resistance to the motion of the gripper's jaws, both disks spin together. But when an object is grasped, the disk attached through the spring slows down, while the directly-connected disk continues at its normal rate. The system's microprocessor can detect that speed difference by counting the slots etched in each disk; that information can be used to determine the force applied as well as the size of the object.

Other approaches use integrated circuits that incorporate a strain gauge and special pressure-sensitive resistive paint. In fact, a simple pressure sensor can be built from the conductive foam that MOS IC's are packed in. That foam has a finite, measurable resistance; when the foam is compressed, resistance decreases. By mounting a piece of foam between two metallic plates, a simple strain gauge could be built.

All those methods of tactile sensing comprise a field of inquiry that is as large as robotics itself. The science of tactile sensing is one in which the home experimenter might achieve a breakthrough by discovering new ways of measuring real-world quantities. So go to it! **R-E**

ROBOTICS

MARK J. ROBILLARD,
ROBOTICS EDITOR

Position sensing

YOU DON'T HAVE TO WORK WITH A robot arm for long to realize that position information is highly important. There are numerous ways of obtaining that sort of information; let's examine several.

Digital position sensing

Most digital approaches to position sensing involve optical devices. Recently in this column we discussed tactile sensing using IR emitters and detectors. Some of the same approaches can be used for position sensing.

For example, a device called an optical shaft encoder or an optical interrupter encodes the position of a shaft by means of opto-electronics. An infrared (IR) detector/emitter pair couples to the shaft of a motor or the pivot of a robot arm and provides a series of digital pulses. The encoder requires only a source of five-volt power. How does it work?

A disk like the one shown in Fig. 1 is used to interrupt a beam of infrared light passing from an LED to a photo-transistor. Overall the disk is transparent, but it has dark stripes that block the IR periodically as it spins. The LED and the transistor are integrated in a single package, as shown in Fig. 2. The encoder disk is attached to the end of a rotating shaft, and then it passes through the slot in the optical interrupter. As the shaft turns, the disk rotates, so the dark areas of the disk periodically prevent the LED light from reaching the photo-transistor. The output of the detector is a series of pulses that may be squared up and fed to digital control circuitry.

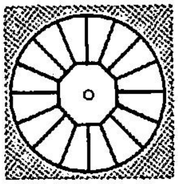


FIG. 1

The disk is manufactured so that the distance between each radial stripe is equal. So the number of pulses that are output indicate how far the shaft has turned. Industrial optical encoders may have several hundred, or a thousand or more divisions.

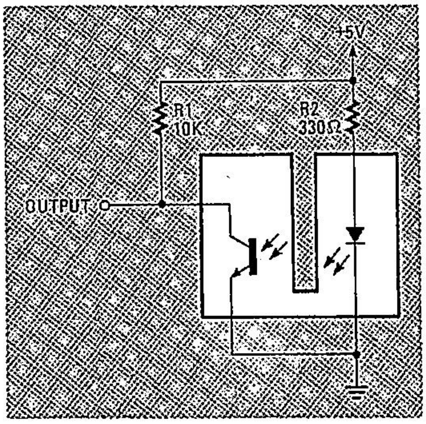


FIG. 2

Commerical encoder disks are usually expensive because they are manufactured under tight mechanical tolerances. In addition, they offer a minimum of 256 slits. For purposes of experimentation, sixteen slits are sufficient because hobbyist-grade motors cannot be positioned very accurately.

You could photocopy the disk in Fig. 1 on a piece of acrylic to provide position information for a Milton Bradley *Robotix* arm. The basic idea is shown in Fig. 3.

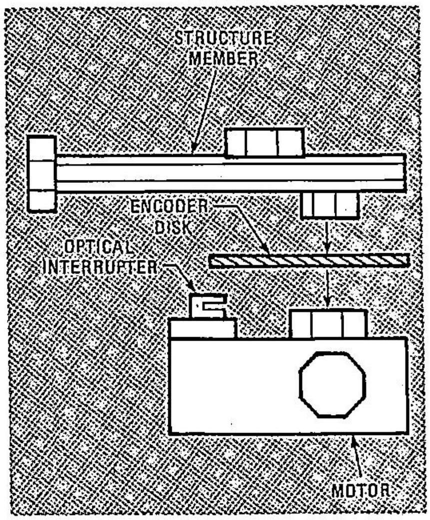


FIG. 3

The optical interrupter is readily available, but if you have trouble locating one, or if you would like to purchase some encoder disks, MJR Digital (Mason Road, Milford, NH 03055) has an experimental kit consisting of nine transmissive encoder disks, nine reflective encoder disks, five H22A1 optical interrupters, and an application note available for $19.95. We'll discuss use of the reflective encoder disks below.

Because two connections go to ground, only three wires must be brought to the encoder. In addition, several encoders may share the ground and five volt lines.

To use the encoder, the computer that controls the motor must monitor the output of the encoder. To begin a motion sequence, the arm must be "homed." In other words, the joint (or joints) in question must be brought to a limit—all the way up, down, left, right, etc.

The computer must then read the status of the encoder. If the output is low, then any subsequent high indicates movement. Conversely, if the initial reading is low, then subsequent highs indicate an absence of movement. It's important to know whether you're starting with a high or a low; otherwise you may not be able to return the arm to the home position accurately.

To move the arm to a specified position, load a counter with the number of slits that must be counted. Then, after homing the arm, turn on its motor and monitor the encoder's output for a pulse. When a high or a low (as previously described) is detected, decrement the internal counter. Repeat that operation until the counter has a value of zero. At that point the arm should be in the desired position.

You could perform the counting in hardware (without a computer) if you like. Doing that is simply a matter of using a counter IC and a logic gate to shut the motor off when the counter reaches zero.

Reflectance decoder

In some situations, a *reflectance-type* encoder disk is more practical. Rather than a clear disk with dark stripes, a reflectance disk is basically reflective with dark stripes. It is used as shown in Fig. 4.

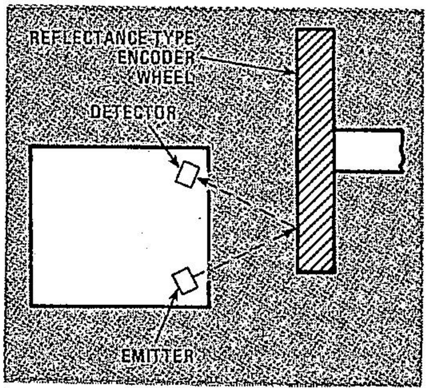


FIG. 4

A beam of infrared light is emitted from an IR emitter at an angle. An IR detector is mounted at a complementary angle. Light will be reflected from the reflective areas of the disk, and not from the striped areas. As the joint moves, the wheel turns, so the output of the detector is a series a highs and lows, like those produced by the transmissive disk. That data can be used to position the arm as previously described.

Analog sensing

Of course, there are other ways to sense the position of an arm. An analog approach might use a variable resistor, an A/D converter, and a little software. The shaft of a potentiometer is connected to the pivot point of a robot's arm, so the potentiometer's resistance should provide an accurate indication of the arm's position. The voltage across that resistor would be read by the A/D converter, and the control computer could then use that information to make an intelligent decision about what step should be done next.

The arm must start from a known position, so, on power up, the arm should be homed. That is best accomplished by rotating each pivot until a microswitch (used as a limit indicator) is activated. At that point motion must halt. Then the control computer should check the output of the A/D device. The voltage read there represents a reference point in relation to which other positions are known.

Some A/D converters require supply voltages of ±12 or even ±15 volts. However, there are several 5-volt A/D IC's on the market. For example, National Semiconductor's A/D0816 can digitize 16 channels of analog information. Each channel has eight bits, so any input between 0 and 5 volts will be converted to one of 256 digital values in steps of about 20 mV. Other A/D IC's provide 12 bits, for a total of 4096 discrete values.

There are several drawbacks to the analog approach. The first is that the potentiometer must be coupled physically to the arm's pivot, and that may be difficult. Also, the mechanical drag of the potentiometer may adversely affect the operation of the arm. And for the beginner, the most serious drawback may be creating the software required to operate the A/D converter.

Speed may also be a problem. Many A/D converters operate much slower than the digital systems controlling them. Often a computer must wait until the A/D converter is ready.

A typical computer-to-A/D dialog might go like this: The computer asks for the current reading, then the A/D works on the request. Some hundreds of microseconds later, the A/D signals the computer that the current reading is ready. Then the computer reads the value. If necessary, the process then repeats.

The speed problem can be alleviated by using a faster A/D converter. However, they're harder to find and more expensive than run-of-the-mill hobbyist-grade devices. And for experimental purposes, a slow A/D converter should prove to be quite sufficient.

As you can see, there are a number of ways of gaining position information about a robot arm. Some of those methods are more useful than others in different circumstances, but the digital approach is generally the simplest to implement as well as the most accurate. R-E

NEW IDEAS

Robot eyes

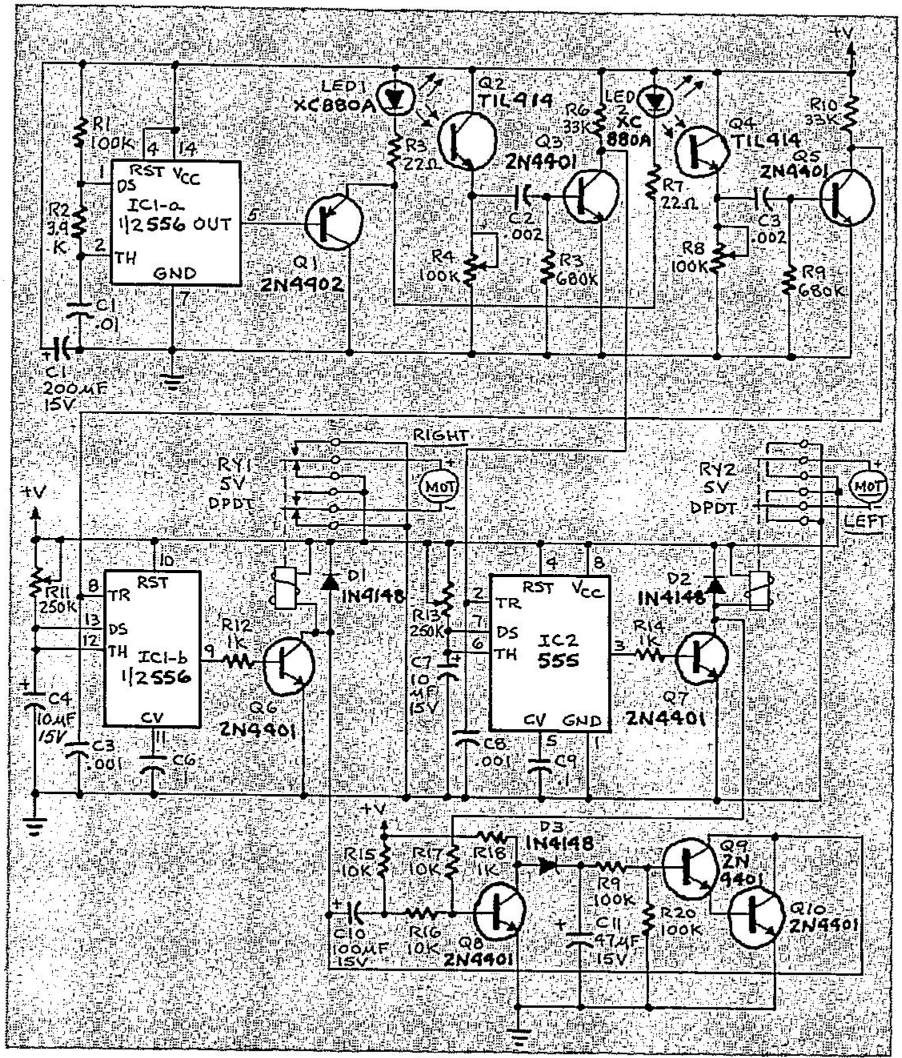


FIG. 1

NEW IDEAS

This column is devoted to new ideas, circuits, device applications, construction techniques, helpful hints, etc.

All published entries, upon publication, will earn $25. In addition, for U.S. residents only, Panavise will donate their *model 333*—The Rapid Assembly Circuit Board Holder, having a retail price of $39.95. It features an eight-position rotating adjustment, indexing at 45-degree increments, and six positive lock positions in the vertical plane, giving you a full ten-inch height adjustment for comfortable working.

I agree to the above terms, and grant **Radio-Electronics** Magazine the right to publish my idea and to subsequently republish my idea in collections or compilations of reprints of similar articles. I declare that the attached idea is my own original material and that its publication does not violate any other copyright. I also declare that this material has not been previously published.

Title of Idea

Signature

Print Name Date

Street

City State Zip

Mail your idea along with this coupon
to: New Ideas Radio Electronics
500-B Bi-County Boulevard
Farmingdale, NY 11735

I HAVE A TOY ROBOT THAT USES TWO small DC motors to move around. The robot was designed to move according to how its joystick was manipulated, but I wanted to make my robot intelligent enough to be able to move about on its own and avoid whatever obstacles happened to appear. I knew that *I* would find it difficult to move around if I had no eyes, so I figured that a pair of "eyes" would help my robot, too. The circuit shown in Fig. 1 represents the fruit of my labor.

shown in Fig. 1 represents the fruit of my labor.

Circuit operation

My robot has one motor mounted under both its left and right sides. The direction the robot moves depends on the direction each motor rotates. For example, the robot can go forward and backward by running both motors in the same direction. And the robot can turn by running one motor forward and the other backward. For example, when the left motor rotates forward, and the right motor moves backward, the robot turns to the right.

It's easy enough to use DPDT relays to control the direction of the motors, but we still need some "smarts." I used an infrared LED and a phototransistor for each eye. The rest of the circuit processes the information provided by those eyes to control the relays.

Half of a 556 timer IC (IC1-a) functions as an astable multivibrator oscillating at a frequency of about 1 KHz. That IC drives transistor Q1, which in turn drives the two infrared LED's, LED1 and LED2. The right eye is composed of LED1 and Q2; those components are mounted side by side—*not* facing each other—about ¼" apart. The left eye is composed of LED2 and Q4, which are mounted like the corresponding parts of the right eye, about four inches away.

If an obstacle appears in front of the right eye, pulses from LED1 are reflected by the obstacle and detected by Q2. The signal from Q2 is amplified by Q3, which triggers IC2, a 555. That IC operates in the monostable mode, and it provides a pulse output with a width of as much as 2.75 seconds, depending on the setting of R11. That pulse output energizes relay RY1, and that reverses the polarity of the voltage applied to the motor. Corresponding portions of the circuit of the left eye operate in the same fashion, using the unused half of the 556 (IC1-b). That action causes the robot to turn away from an obstacle.

When an obstacle appears in front of both eyes, both relays will be activated, so the robot will back up. The circuit composed of Q8–Q10 (and associated components) provides additional "on" time for the right motor. That helps the robot avoid getting trapped in a narrow passage.

Construction

Construction of the circuit is not critical, so feel free to use the technique you prefer. Just be careful with the orientation of polarized components and semiconductors. The circuit can operate from any voltage between 4.75 and 7.5 volts. Potentiometers R4 and R8 adjust the sensitivity of the phototransistors; you might adjust them to respond to an obstacle that is twelve inches away. Potentiometers R11 and R13 control the amount of time the motors will be reversed. That will depend partly on the surface your robot is traversing. Too little time on a rough surface might not affect direction at all, and too much time could cause constant overshooting. You'll have to experiment a little.—*John Ellis*

NEW IDEAS

Light-seeking robot

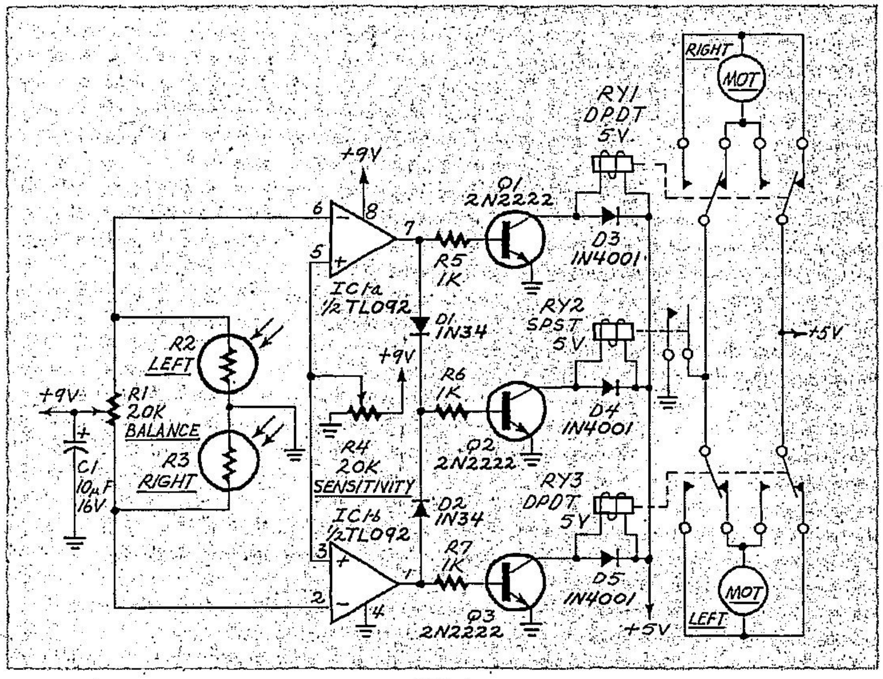


FIG. 1

HERE'S A SIMPLE CIRCUIT FOR DEMONstrating some basic principles of robotics. The circuit is light seeking; with careful adjustment it will follow a flashlight around a darkened room. The heart of the circuit is a pair of photocells that determine the direction in which the robot will move. Depending on how light falls on the cells, the robot can be made to move forward, left, or right. Motion is provided by two independent DC motors; a swivel-wheel maintains balance.

How it works

In the circuit shown in Fig. 1, each photocell is connected to an op-amp configured as a comparator. When sufficient light falls on photocell R2, the voltage at the inverting input (pin 6) of IC1-a will fall below the voltage at the noninverting input (pin 5), so the output of the comparator will go high, and transistors Q1 and Q2 will turn on. That will enable relays RY1 and RY2, and thereby provide power for the right motor. The robot will then turn left.

Likewise, when light falling on R3 lowers its resistance, Q2 and Q3 will turn on, the left motor will energize, and the robot will turn right.

Notice that diodes D1 and D2 provide an OR function: Q2 turns on when either comparator goes high (or when both do). When Q2 turns on, it energizes relay RY2, which provides a path to ground for either motor coil, or both. When there is not enough light falling on the photocells, neither comparator will go high, so RY2 will be off. That is what prevents the robot from moving when neither photocell is lit.

Also note that the photocells and motors are cross-wired—the left photocell controls the right motor, and the right photocell controls the left motor.

NEW IDEAS

This column is devoted to new ideas, circuits, device applications, construction techniques, helpful hints, etc.

All published entries, upon publication, will earn $25. In addition, for U.S. residents only, Panavise will donate their *model 333*—The Rapid Assembly Circuit Board Holder, having a retail price of $39.95. It features an eight-position rotating adjustment, indexing at 45-degree increments, and six positive lock positions in the vertical plane, giving you a full ten-inch height adjustment for comfortable working.

I agree to the above terms, and grant **Radio-Electronics** Magazine the right to publish my idea and to subsequently republish my idea in collections or compilations of reprints of similar articles. I declare that the attached idea is my own original material and that its publication does not violate any other copyright. I also declare that this material has not been previously published.

Title of Idea

Signature

Print Name Date

Street

City State Zip

Mail your idea along with this coupon
to: **New Ideas Radio-Electronics,**
500-B Bi-County Boulevard
Farmingdale, NY 11735

Construction

Building this circuit requires more mechanical than electronic skills. In addition to a circuit board, you'll have to provide a platform for the motors, the swivel wheel, and the batteries. You could generate the five-volt motor/relay supply from a six-volt battery or gel-cell in series with a power diode. Make sure to use a diode that can handle about four times the maximum current drain of a single motor. To provide noise immunity, the comparators should be powered from a separate low-current nine-volt supply.

The photocells should be mounted on the front of the platform, several inches apart, with a light barrier between them. You'll have to experiment to find the exact dimensions for your application, but try mounting the photocells two inches apart, and make the light barrier about four inches long. Also, it's a good idea to coat the barrier and whatever the photocells are mounted on with flat black paint.

After wiring the circuit, check it over and correct any errors. Then turn off the lights and adjust R4 so that the robot doesn't move when only ambient light is falling on the photocells. Next adjust R1 so that the photocells are equally responsive to your signal light. Last, try leading the robot around the room. —*J. A. Tavares*