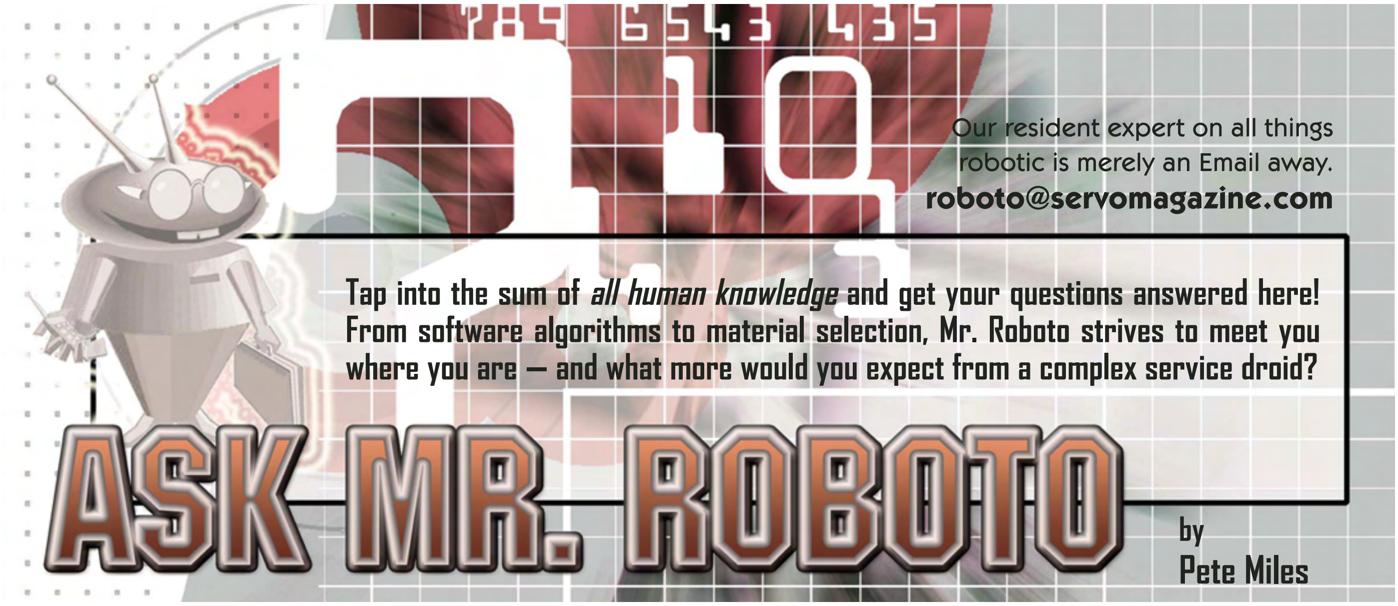


Our resident expert on all things robotic is merely an Email away.

roboto@servomagazine.com

Tap into the sum of *all human knowledge* and get your questions answered here! From software algorithms to material selection, Mr. Roboto strives to meet you where you are — and what more would you expect from a complex service droid?

ASK MR. ROBOTO

by
Pete Miles

Q. I have a mini sumo that uses three of the long range Sharp sensors — the GP2Y0D02YK. I have the sensors mounted to the sides of my robot and one at the front. Basically, my robot will attack anything the front sensor sees, and when the side sensors see a robot, it will spin around until the front sensor sees the other robot. Then it will attack that robot. The problem I am having is that when I am at home, the robot sees my hand with no problem and will always go after it. But when I go to tournaments, most of the time my robot doesn't see the other robot, and ends up spinning in a circle looking for the other robot. Do you have any suggestions that could help me?

— Steve Nations

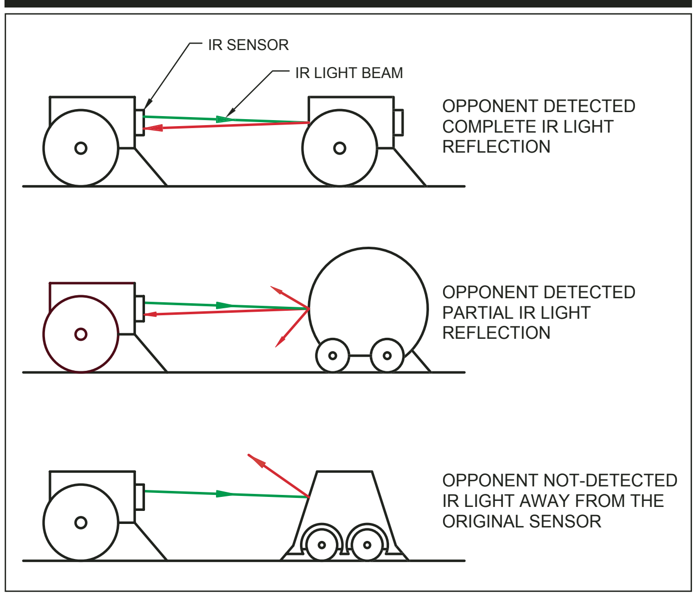
A. The Sharp GP2Y0D02YK Long Distance Measuring Sensors are one of my favorite sensors to use in any robot. They are quite reliable and not very prone to surrounding light interference. I have had these sensors detect as small as a 0.020 inch diameter wire, two feet away from the sensor. But there are a couple things that you need to keep in mind when using these sensors to field them effectively.

First off, these sensors are reflective sensors, and they require a portion of the original narrow beam of infrared light to be reflected right back at the sensor, or it doesn't "see" or sense anything. Your hand is one of the better surfaces for reflecting light back to the sensor. It has many irregular surface, wrinkles, creases, and folds that almost ensures that you will be reflecting some portion of the IR light back at the sensor. But a robot has a lot of flat surfaces that tend to reflect the light away from the sensor, not back to the sensor.

To illustrate this, place a clean mirror in front of the sensor. If the orientation of the mirror is such that all of the IR light is reflected back to the sensor, then the sensor will always detect the mirror. But if you start angling the mirror upwards, or to the sides, the sensor will stop "seeing" the mirror that is right in front of it. The closer the mirror is to the sensor, the greater the angle (which is only a few degrees) that the mirror needs to be misaligned for the sensor to stop seeing the mirror. What gets worse is that as the distance between the mirror and the sensor increases, the more precise the angular orientation must be in order for the sensor to detect the mirror. Past two feet, the mirror has to be almost perfectly aligned in order for the sensor to see the mirror. Figure 1 shows a simple illustration of this.

Fortunately, these sensors are very sensitive, so that any imperfection on the surface of the target is more likely to cause light to be reflected back to the sensor. An easy way to observe this is to conduct the same tests with a piece of flat metal or plastic. You will find that you can have greater angles on the target surface for the sensor to see it. The rougher the surface, the greater the angular misalignment can be for detecting the target.

Figure 1. Illustration of angular refection for IR sensors.



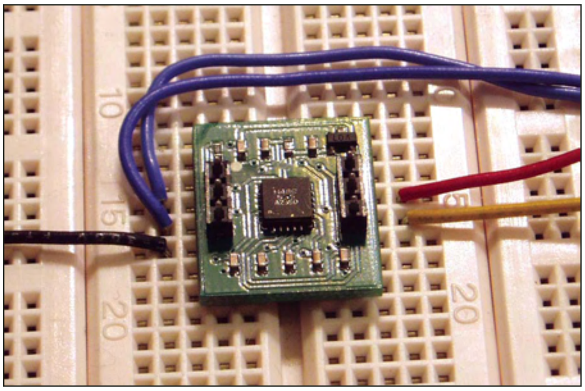


Figure 2. Hitachi H48C three-axis acceleration module from Parallax, Inc., mounted on a breadboard.

Now how does this relate to a mini sumo? Well, they have lots of little parts and edges that do a good job at reflecting IR light back to the sensor. But the angles of the surfaces on all these little parts — with respect to your robot — is constantly changing. So there will be times that your sensors will get a good strong return signal, and there will be other times when the other robot is right in front of yours, and the sensors don't see it at all.

So it is always better to test your robot tracking sensors against other robots to see how well they work. This often overlooked flaw in using reflective sensors can also be used to your advantage. If you place angled plates on the sides of your robot, you can force your opponent's IR sensors to reflect up in the air. This will, in effect, make your robot invisible to the other robots.

The second problem with these sensors is the ~50 ms time delay between each measurement. These sensors work by sampling a certain number of return signals to determine if the input is real before outputting the result. This sampling period takes about 40 ms, and there is about a 10 ms delay before an output occurs. This means that the other robot must be in the sensor's field-of-view for about 40 ms and returning a lot of reflected IR light back to the sensor, or it won't detect the robot.

If the robots are moving slowly, then this usually isn't a real problem. But in the case of the spinning robot/sensor, then the amount of time the sensor's IR light is on the opponent will be much less than if the robot is driving past your robot on its own. The faster your robot is spinning, the less likely the sensors will see the robot. Not only is there less time for the sensor's IR light to be on the other robot, but the angular reflections will be worse because the IR light beam is traveling in an arc.

The approach that you are using in your robot to spin around and stop when the forward sensors see its opponent is a good one, and I have used it myself. What you need to do is slow down the spin rate, and add a spin time delay that will stop the robot from spinning if it doesn't detect an opponent. For example, you could set the time delay to limit the robot to spinning only 90 degrees, or one complete revolution before continuing to move forward.

With all this being said, you should be able to improve your mini sumo robot's performance with a little bit of software adjustments by limiting how much the robot spins in a circle, and reducing the speed in which the robot spins. Hopefully this helps you.

Q. Do you know of any good sensors that can determine if your robot is upside down or on its sides so that it can automatically flip itself back upright?

— Lee Banks

A. Parallax, Inc. (**www.parallax.com**), has a new three-axis accelerometer module that should work well for your application. This module uses a single Hitachi (**www.hitachi.com**) H48C three-Axis accelerometer that is capable of measuring ±3gs of acceleration on each axis (an example code is available on the *SERVO* website at **www.servomagazine.com**). The module includes a built-in 3.3 voltage regulator so that the module can be used with standard five-volt circuits, and it uses the Microchip (**www.microchip.com**) MCP3204 four-channel 12-bit analog-to-digital (ADC) converter with SPI serial interface. The MCP3204 ADC uses the SPI serial interface protocol that is compatible with the SHIFTIN and SHIFTOUT commands on a BASIC Stamp.

The module is relatively small, only 0.7 inches by 0.8 inches in size, and its six I/O pins can be mounted on any standard breadboard. Figure 2 shows a photograph of the module. When the sensor is not moving, it can measure the gravitational pull of the Earth, and the values read on the x, y, and z axis of the sensor will tell you the orientation of the accelerometer with respect to the Earth. By knowing the orientation of the accelerometer on the robot, you can determine the orientation of the robot with respect to the ground.

What makes this sensor attractive is that it can measure all three axis at the same time. Most other low-cost accelerometers only measure two axis, and a robot would require a pair of these sensors to obtain 3D information. In addition, the circuit board for the other accelerometers must have a vertical extension for mounting the second accelerometer, and making sure that the two accelerometers are at 90 degrees with respect to one another is difficult.

The following formula is used to determine the acceleration on each of axis of the accelerometer, where *Gaxis* is the acceleration in gs on a particular axis, *Axis*, and *Vref* is a self calibration reference voltage.

$$G_{axis} = \frac{Axis - Vref}{366.3\ mV/g}$$

Since the MCP3204 converts the analog voltage from the H48C into a 12 bit binary number, the microcontroller must convert this back into a number that is easily understood. The following formula shows how the 12 bit counts for Axis and Vref are converted into a numerical number. The 4095 is the maximum number of counts for a 12 bit number,

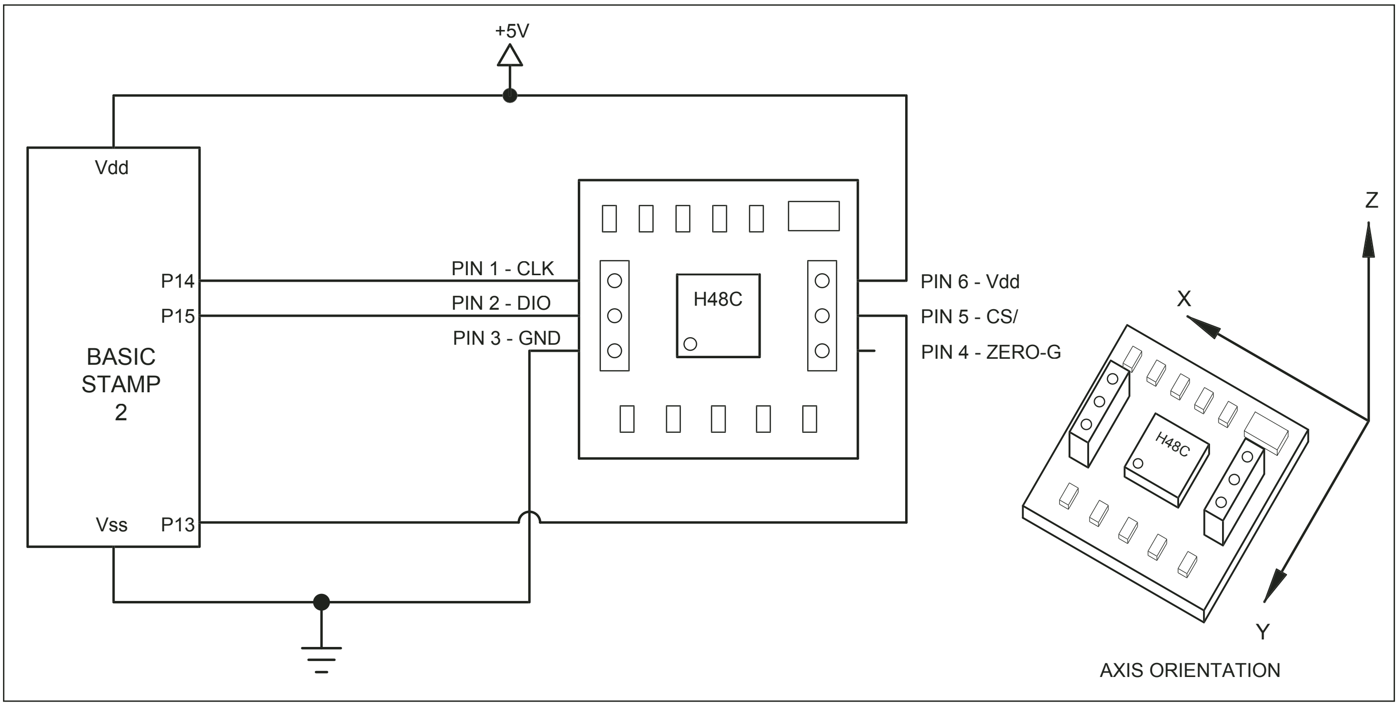


Figure 3. Wiring the Hitachi H48C acceleration module to a BASIC Stamp.

and 3.3 V represents the maximum voltage span.

$$G_{axis} = \frac{Axis - Vref}{366.3\,mV/g} * \left(\frac{3.3V}{4095}\right)$$

Figure 3 shows a simple schematic for hooking up this sensor to a BASIC Stamp, and the example programs shown here illustrate how to read the acceleration values. Once the acceleration values have been read in for all three axes, the Orientation subroutine is called in this example program. Here this routine assumes that the Z-axis of the sensor is pointing up towards the sky when the robot is on flat level ground, and the Y-axis is pointing towards the front of the robot.

Figure 3 also shows the orientation of the three axes on the accelerometer module. Right now, this routine will tell you if the robot is vertical or inverted (i.e., upside down) and whether the robot is leaning forward and/or to the sides. In your application, you would want to add some threshold values to tell the robot to execute corrective actions if the threshold values are exceeded. For example, if the robot is leaning to the side by more than 45 degrees, chances are that is about to fall over, thus it should start turning towards that side to prevent rolling over. **SV**