CS148
Building Intelligent Robots

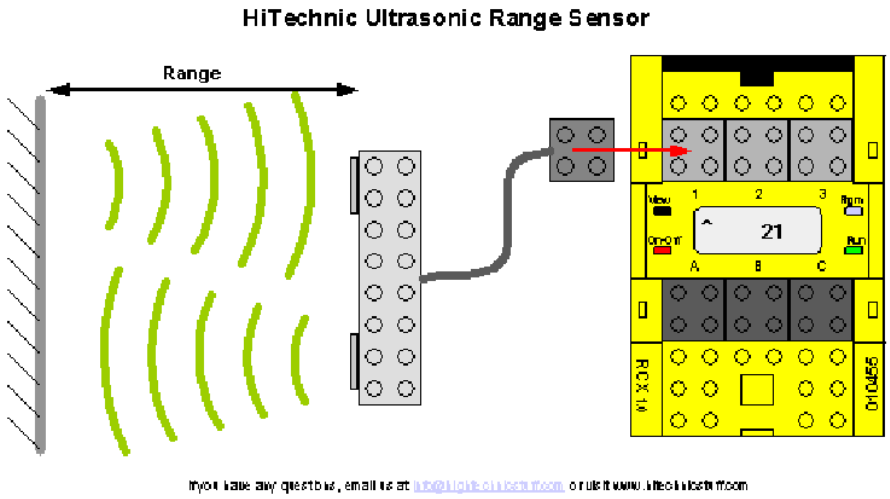
Week 2 : Subsumption Architecture and Sonar *Out: 20 Sept 2005*

Preliminary Tasks *before 22 Sept 2005, 9am*

The purpose of this week's lab is to introduce the subsumption architecture and sonar sensors. For this assignment your robot will need a sonar sensor and a light sensor. *Hint: You might want to use your light calibration from the previous lab.*

Sonar Review

Ultrasonic sensors are often used in robots for obstacle avoidance, navigation and map building. Much of the early work was based on a device developed by Polaroid for camera range finding. From the Hitechnic Ultrasonic Sensor web page we learn that their "ultrasonic range sensor works by emitting a short burst of 40kHz ultrasonic sound from a piezoelectric transducer. A small amount of sound energy is reflected by objects in front of the device and returned to the detector, another piezoelectric transducer. The receiver amplifier sends these reflected signals (echoes) to [a] microcontroller which times them to determine how far away the objects are, by using the speed of sound in air. The calculated range is then converted to a constant current signal and sent to the RCX." The Hitechnic sensor is different from the Polaroid sensor in that it has separate transmitter and receiver components while the Polaroid sensor combines both in a single piezoelectric transceiver; however, the basic operation is the same in both devices.



Lab 2: Subsumption Architecture and Sonar *22 Sept 2005*

There are a number of complications involved in interpreting the time-of-flight information returned by an ultrasonic sensor. If the sensor face is parallel to the surface of the nearest object and that surface is flat, reflective and relatively large, e.g., a plaster wall, then the information returned by the sensor can be reasonably interpreted as the distance to the nearest object in front of the sfnsor. However if the object deviates significantly from this ideal object, the time-of-flight information can be misleading. Here is one of the more benign sorts of interpretation error caused by the fact that the signal (corresponding to a propagating wave of acoustic energy) spreads as it propagates further from the sensor with most of the energy of the leading edge confined to a 30 degree cone. If the surface is angled with respect to the face of the sensor (as it is below) then the time of flight information will record the distance to nearest point within the 30-degree cone.

Use the sonar the same way you would use a light sensor. **When attaching the connector plate to the input port on the RCX, make sure the plate is oriented so that the cable is on the left**

0.1

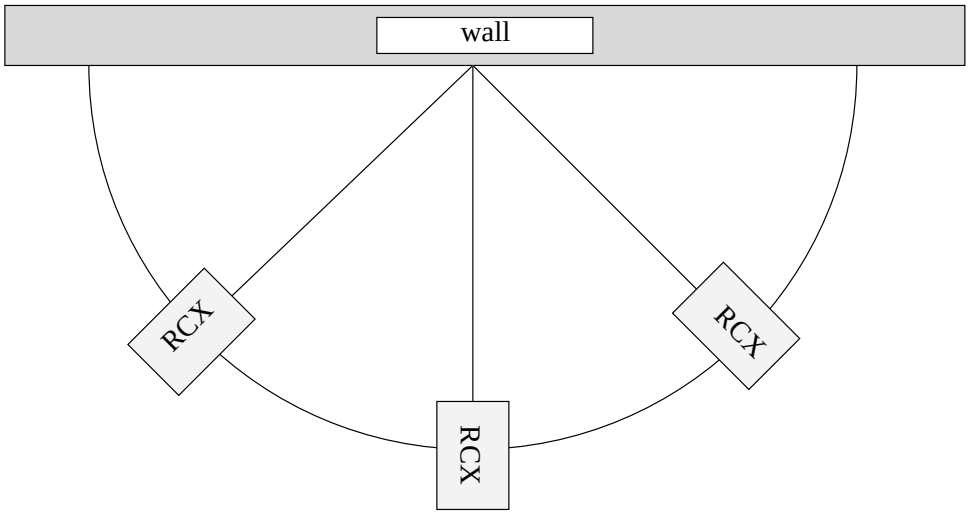
Task:

- You first should to understand how the sonar behaves when facing a small surface. Take a small, flat object, and place it approximately 15 inches away from your RCX so that the surface of the object is perpendicular to the sonar. Record the sonar measurements as you rotate the object. Take measurements in 15 degree increments and make a rough plot. If your measurements fluctuate rapidly, then average them out.

0.2

Task:

- Now see how the sonar behaves when it is facing a large flat surface. Pick a point on the wall and imagine a semicircle around that point. Now place your RCX at various points on that semicircle, always facing the same point on the wall. Record the sonar readings you get. Make a rough plot.



0.3

In the subsumption architecture, your robot will need to implement three behaviors. Your task will be to write an arbitration module and two of the behavior modules. The first module, *wander*, which will be provided for you, is responsible for having the robot wander randomly in its environment. The second module, *collision avoidance*, should monitor the robot's area using sonar and perform an evasive action to avoid collisions. The third module, *halt*, will stop the robot and play a tune if it detects a light brighter than the ambient light conditions. Finally, the *arbitrator* module should control the outputs so that the robot behaves appropriately.

You will be given a file containing the wander module, to which you should add a collision avoidance, halt, and arbitrator module. This module should abstract out the motor control for the robot and should subsume the modules in the following order:

- Highest priority: *halt*
- Medium priority: *collision avoidance*
- Lowest priority: *wander*

You can copy the stencil code by typing this in your working directory (from a cygwin shell)

```
cp -r l:/asgn/2005/lab2/* .
```

Task:

- Write the collision avoidance, halt, and arbitration modules so that your robot will perform the behaviors in a prioritized manner as described above. Make sure your LCD will output the name of the process that currently has the highest priority.