

Measuring robot distance to a surface using Ultrasonic sensors

Ultrasonic sensors are a common way to find the distance from a robot to the nearest surface

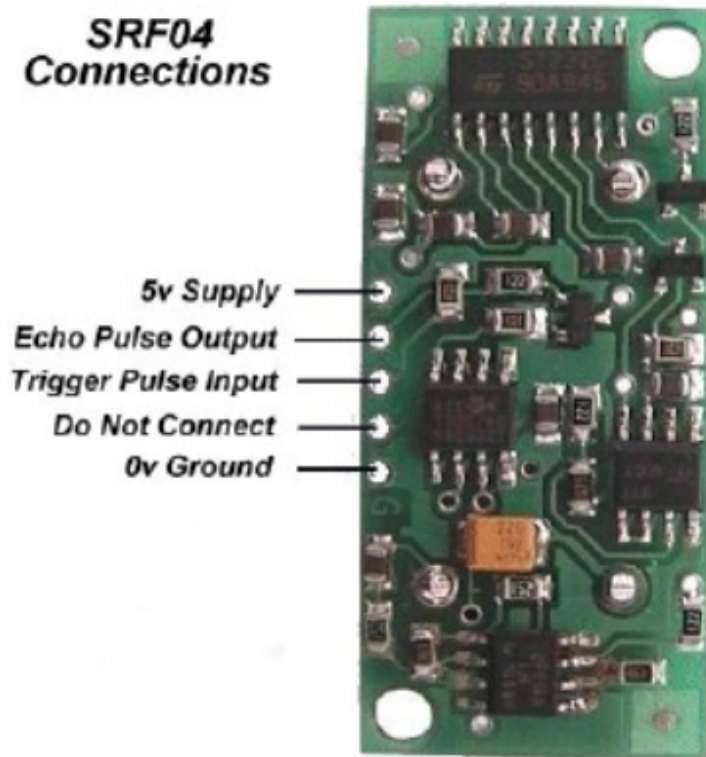
Ultrasonic rangefinders

Ultrasonic rangefinders use the travel time of an ultrasonic pulse to determine distance to the nearest object within the sensing cone. There are a variety of different ways that various ultrasonic sensors communicate the measurement result including:

- Ping-Response (ex. [Devantech SRF04](#), [VEX Ultrasonic Rangefinder](#))
- Analog (ex. [Maxbotix LV-MaxSonar-EZ1](#))
- I2C (ex. [Maxbotix I2CXL-MaxSonar-EZ2](#))

Measuring robot distance to a surface using Ultrasonic sensors

Ping-Response Ultrasonic sensors



To aid in the use of Ping-Response Ultrasonic sensors such as the Devantech SRF04 pictured above, WPILib contains an Ultrasonic class. This type of sensor has two transducers, a speaker that sends a burst of ultrasonic sound, and a microphone that listens for the sound to be reflected off of a nearby object. It requires two connections to the cRIO, one that initiates the ping and the other that tells when the sound is received. The Ultrasonic object measures the time between the transmission and the reception of the echo.

Creating an Ultrasonic object

```
Ultrasonic ultra(ULTRASONIC_ECHO_PULSE_OUTPUT,  
ULTRASONIC_TRIGGER_PULSE_INPUT);
```

Example 4: C++ example of creating an ultrasonic rangefinder object

```
Ultrasonic ultra = new Ultrasonic(ULTRASONIC_ECHO_PULSE_OUTPUT,  
ULTRASONIC_TRIGGER_PULSE_INPUT);
```

Example 5: Java example of creating an ultrasonic rangefinder object.

Measuring robot distance to a surface using Ultrasonic sensors

Both the Echo Pulse Output and the Trigger Pulse Input have to be connected to digital I/O ports on a Digital Sidecar. When creating the Ultrasonic object, specify which channels it is connected to in the constructor, as shown in the examples above. In this case, `ULTRASONIC_ECHO_PULSE_OUTPUT` and `ULTRASONIC_TRIGGER_PULSE_INPUT` are two constants that are defined to be the digital I/O port numbers. Do not use the ultrasonic class for ultrasonic rangefinders that do not have these connections. Instead, use the appropriate class for the sensor, such as an `AnalogChannel` object for an ultrasonic sensor that returns the range as an analog voltage.

Reading the distance

```
Ultrasonic ultra(ULTRASONIC_PING, ULTRASONIC_ECHO);  
ultra.SetAutomaticMode(true);  
int range = ultra.GetRangeInches();
```

Example 6: C++ example of creating an ultrasonic sensor object in automatic mode and getting the range.

```
Ultrasonic ultra = new Ultrasonic(ULTRASONIC_PING, ULTRASONIC_ECHO);  
ultra.setAutomaticMode(true);  
int range = ultra.getRangeInches();
```

Example 7: Java example of creating an ultrasonic sensor object in automatic mode and getting the range.

The following two examples read the range on an ultrasonic sensor connected to the output port `ULTRASONIC_PING` and the input port `ULTRASONIC_ECHO`.

Analog Rangefinders

Many ultrasonic rangefinders return the range as an analog voltage. To get the distance you multiply the analog voltage by the sensitivity or scale factor (typically in inches/V or inches/mV). To use this type of sensor with WPILib you can either create it as an `AnalogChannel` and perform the scaling directly in your robot code, or you can write a class that will perform the scaling for you each time you request a measurement.

I2C and other Digital Rangefinders

Rangefinders that communicate digitally over I2C, SPI, or Serial may also be used with the cRIO though no specific classes for these devices are provided through WPILib. Use the appropriate communication class based on the bus in use and refer to the datasheet for the part to determine what data or requests to send the device and what format the received data will be in.