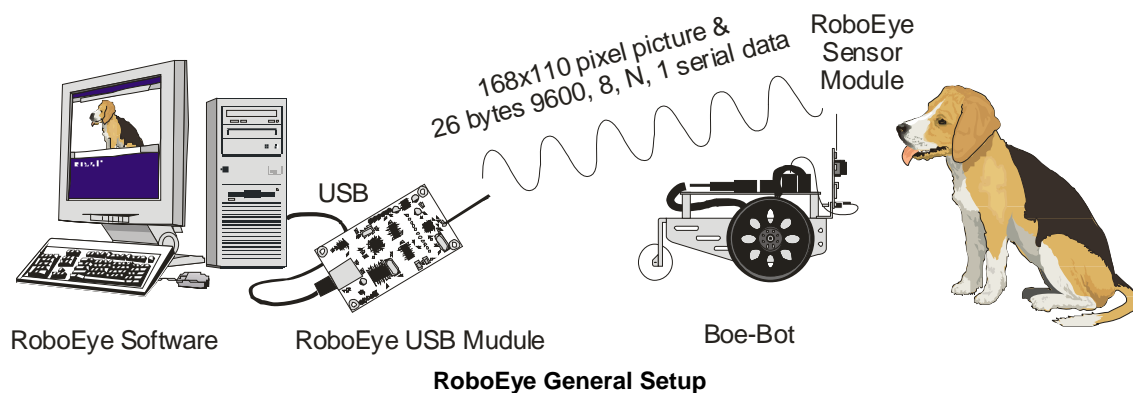


## RoboEye Wireless Vision Sensor (#30008)

RoboEye provides both a simple and sophisticated method of robotics vision and control. In its simple form, the RoboEye Sensor sends a 168x120 pixel image over a 2.4 GHz wireless band for display on your PC using the RoboEye software. A additional serial I/O line may be used to send control commands or data between the RoboEye software terminal and a BASIC Stamp. The wireless data link includes built-in error checking.

In its more advanced form, software developers can take advantage of RoboEye's open-sourced image transmission protocol for further decision making and analysis on a computer. This is done using an OCX application and intercepting a data packet stream which describes pixel colors and location. This type of use is similar to the popular CMU Cam, except that image analysis can be done with PC software instead of on the BASIC Stamp.

Pay attention to the setup process described in this documentation. Installation of the USB driver, setup of the RoboEye Sensor Module and configuration works best if done in a particular order.



## Packing List

Verify that your package is complete and contact us if anything is missing. This kit also includes the hardware you will need for a basic connection to the Parallax Boe-Bot.

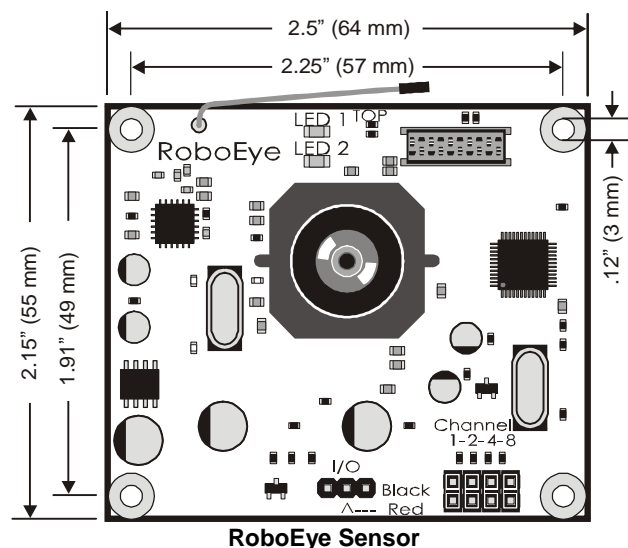
- (1) RoboEye Sensor Module (#727-30008)
- (1) RoboEye USB Module (#727-30008)
- (1) USB Cable (#need part number from Kelly)
- (1) 3-pin 10" female/female cable (white, red, black) (#805-00001)
- (1) Lens and lens holder (black plastic) (#721-30008)
- (2) Right-angle aluminum bracket parts (#720-28215)
- (4) 4/40 1/4" screws (#700-00028)
- (4) 4/40 nuts (#700-00003)
- (2) M2 x 10 screws (#need part number from Kelly)

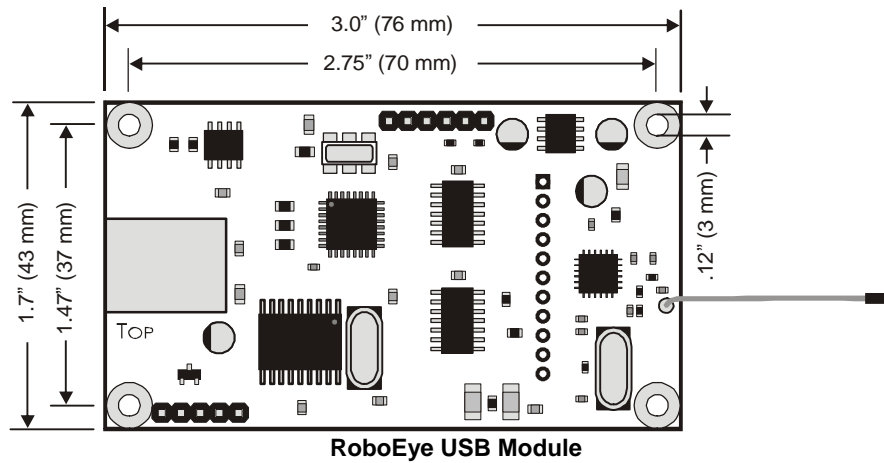
RoboEye PC software and BASIC Stamp source code is available for download from our RoboEye web page on the Parallax web site ([http://www.parallax.com/detail.asp?product\\_id=30008](http://www.parallax.com/detail.asp?product_id=30008)).

## Electrical Specifications

<b>Power supply</b>	Roboeye Sensor 4-9V (5 recommended); RoboEye USB Module (USB port provides 5V and 100 mA to run the hardware)
<b>RoboEye Sensor Supply Current</b>	50 mA (typical)
<b>I/O Line Voltages</b>	Maximum 5V input voltage; 3V output voltage (logical high); 0.3V output voltage (logical low)
<b>Serial Communication</b>	UART, 9600 Baud, 8 bit data, 1 stop bit with 26 byte buffer
<b>Frequency</b>	2400 – 2524 MHz with 16 different channels (jumper configured)

## Physical Dimensions





## Lens and Lens Holder Mounting to RoboEye Sensor Module



1. Locate the (2) M2 x 10 screws, RoboEye Sensor Module and Lens/Lens Holder assembly.



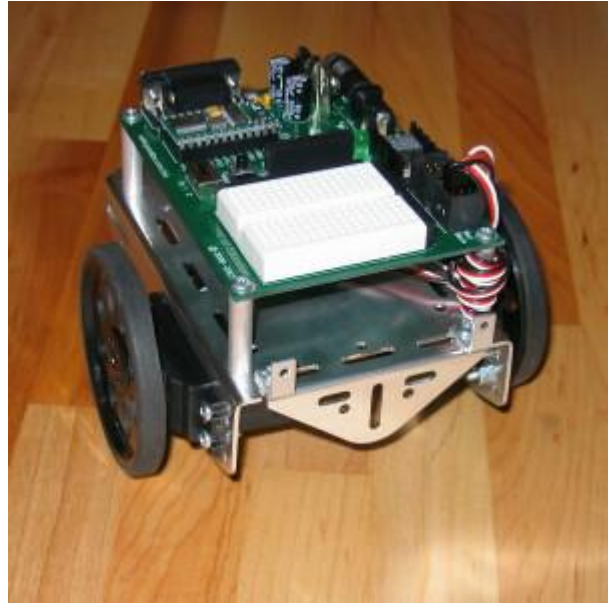
2. Thread the M2 x 10 screws through the back of the printed circuit board, into the lens/lens holder assembly. No nuts are required since the screws are self-threading.

## Mount the RoboEye Sensor Module to a Boe-Bot

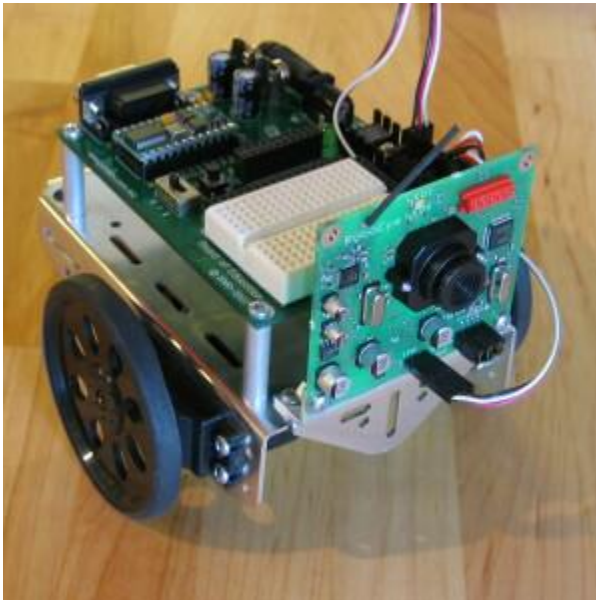
To mount the RoboEye Sensor Module on a Boe-Bot follow these additional steps. Disconnect power from the Board of Education before you begin.



1. Locate the (4) 4/40 1/4" screws, (4) 4/40 nuts, (2) right-angle brackets, and the 10" female/female three-pin cable.



2. Using (2) 4/40 1/4" screws and (2) 4/40 nuts, mount the (2) right-angle brackets to the Boe-Bot's front slots.



3. Using (2) 4/40 1/4" screws and (2) 4/40 nuts mount the RoboEye Sensor Module to the brackets. Connect the 3-pin cable between the RoboEye Sensor Module and the BOE's P15 servo port, paying particular attention to polarity.

## USB Driver Installation

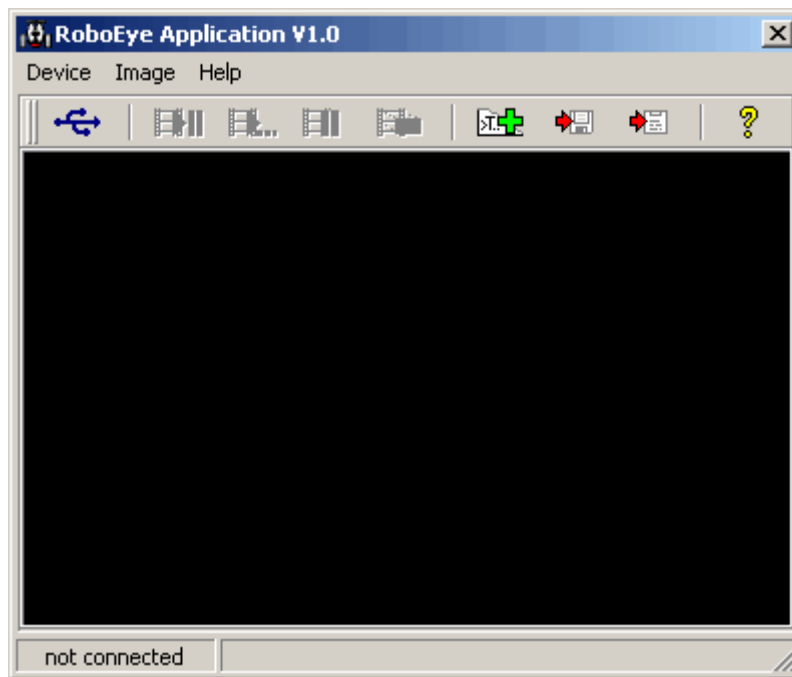
The USB drivers are installed first.

1. Go to the Parallax RoboEye web page [http://www.parallax.com/detail.asp?product\\_id=30008](http://www.parallax.com/detail.asp?product_id=30008) and download "RoboEye Software and Drivers". This is a zipped software package.
2. Extract the contents of this zip file to your computer so you can access the files.
3. Connect the RoboEye USB Module to the PC using the supplied USB cable.
4. The PC will report that new hardware has been found. Point the PC to the FTDI drivers unzipped to your PC.

The RoboEye control software consists of three parts: CamViewXControl.ocx, RoboEyeInstall.exe and RoboEye.exe. All of these files are installed by running RoboEyeInstall.exe. Run RoboEyeInstall.exe.

## RoboEye Software Setup

1. Run RoboEyeSetup.exe software. Then run RoboEye.exe.



The icons provide the following control:



connect USB interface



double the picture size



capture a single picture



opens terminal



start repeat capturing



save picture to disk

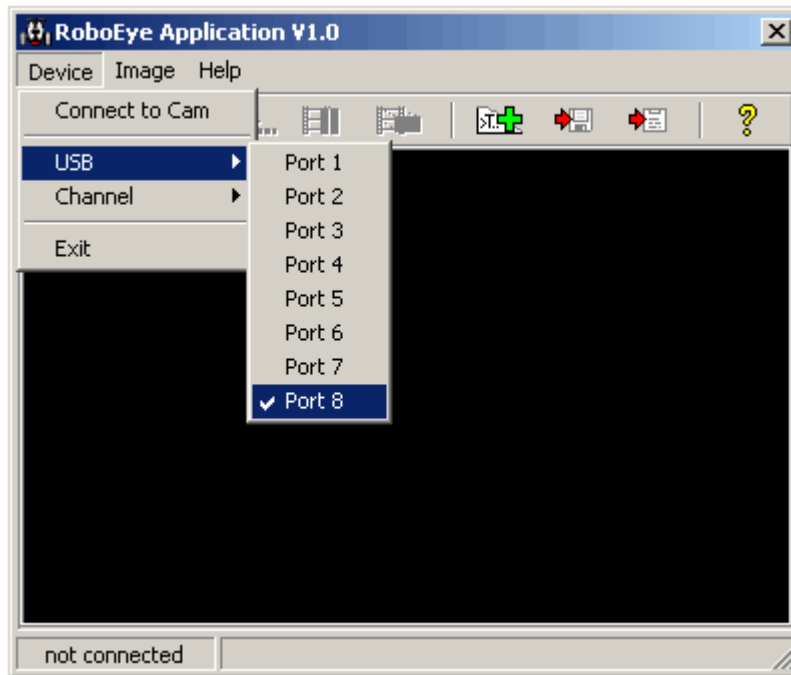


stops capturing

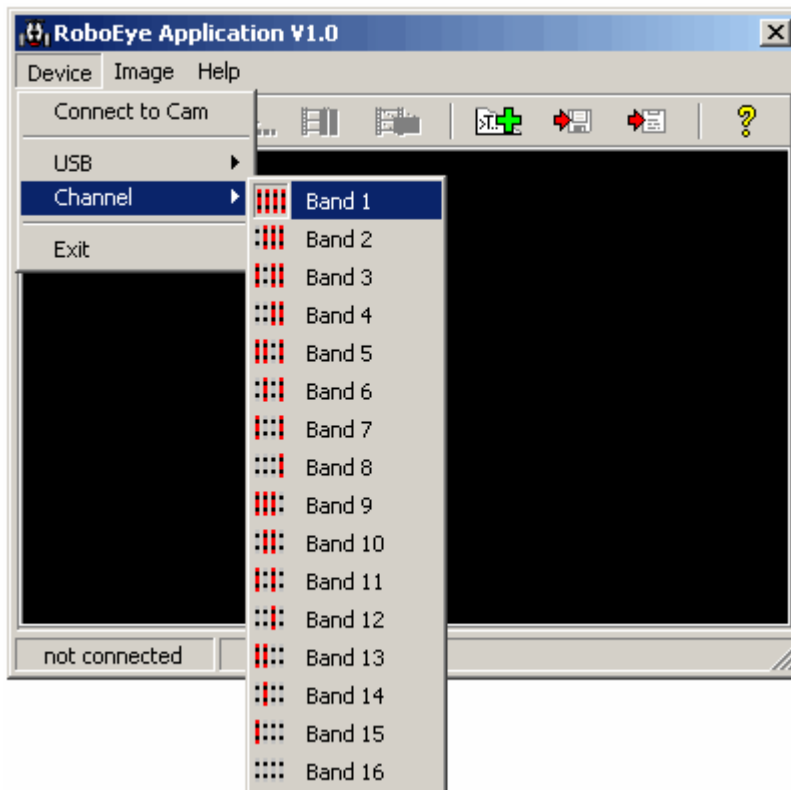


copy picture to clipboard

2. Choose the appropriate USB port in the RoboEye software. This must match the USB virtual COM port assigned by the Windows Device Manager.



3. Select the wireless transmitting channel, based on the placement of jumpers on the RoboEye Sensor Module.



4. Apply power to the remote RoboEye Sensor Module. After power is applied, the green LED2 flashes.
5. Click the green "+" to show the terminal window.
6. Click on the "Connect to camera" icon under the Device menu. Now you can test the connection between the RoboEye Sensor Module and the RoboEye USB Module connected to the PC.
7. In the white terminal window, type "ping" and press the >>> to send the data. RoboEye Sensor Module will respond with the firmware revision number.



Real-time moving picture transmission is not possible with RoboEye. It is also impossible to store a complete picture on board the picture sensor. Sharp and clear pictures are obtained using a fixed camera position on a fixed scenery.

## Serial Communication Between RoboEye Sensor Module and PC

### Sending Commands to the BASIC Stamp

Open the RoboEye software terminal window (press the open terminal button) to get access to the serial I/O line. The serial I/O line is configured as a UART with 9600 Baud (9600 Baud, one stop bit, no parity, no hardware handshake) in half duplex mode. This means that command strings are sent directly from the terminal input line through the RoboEye USB Module to the RoboEye Sensor Module.

For example, if you want to transmit the string "Hello", type the individual characters and press <Enter> or click on the send button (>>>) at the end of the line with the mouse. The five characters are packed into a message and will be transmitted to the picture sensor wirelessly. No special data transmission protocol exists.

Transmission between the PC and RoboEye Sensor Module is most reliable if the picture auto-capture is turned off. It is also important to note that hitting the send button (>>>) sends the entire string of characters present in the white pane.

### Sending Data from the BASIC Stamp to the PC

Each incoming message will be buffered until a <CR> is detected. Incoming messages are limited to 26 bytes. Here is a very simple example using the BASIC Stamp 2 module:

```
' -----[ Title ]-----
' BASIC Stamp to the PC Basic Communication - RoboEyeSimpleTransmission.BS2

' {$STAMP BS2}                                ' Stamp directive
' {$PBASIC 2.5}                                ' PBASIC directive

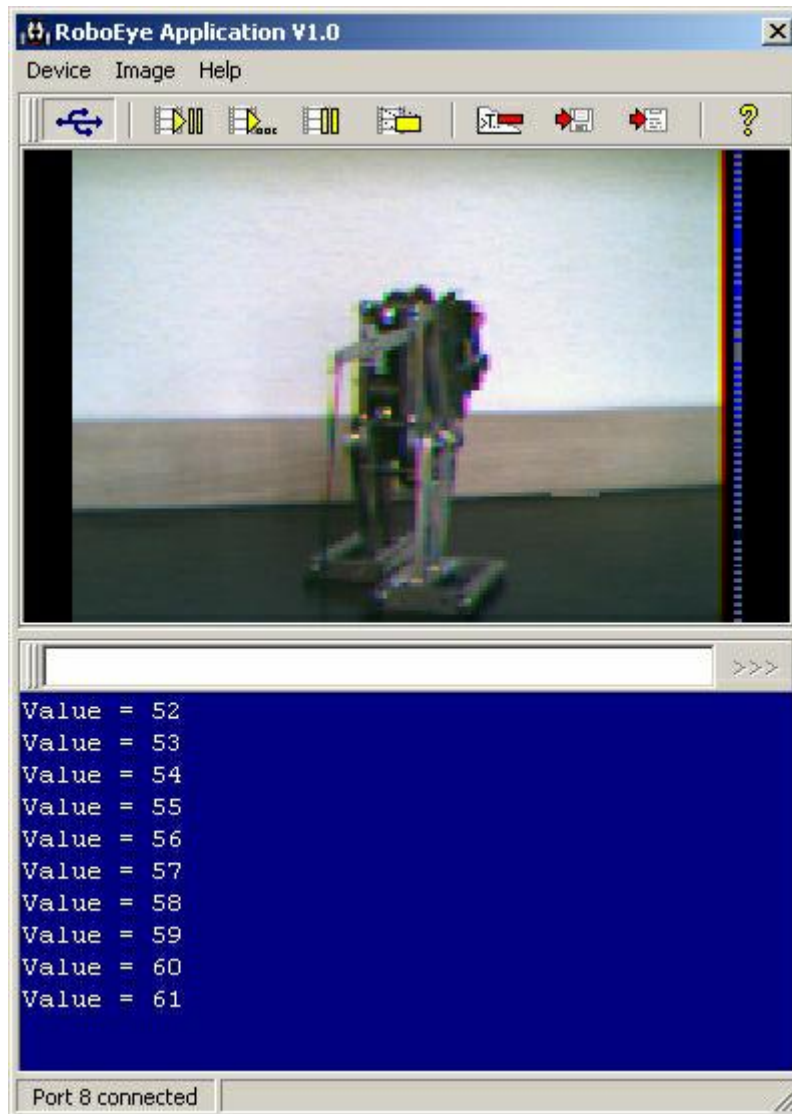
' -----[ Variables ]-----

counter          VAR      Byte                  ' FOR...NEXT loop counter

' -----[ Main Routine ]-----

controlPanel:
  FOR counter = 1 TO 256
    SEROUT 15, 84, ["Value = ",DEC counter, CR]
    PAUSE 1000
  NEXT
END
```

The example above sends the value "counter" to the BASIC Stamp P15 with 9600 Baud (see the SEROUT command in the BASIC Stamp User's Manual).



**RoboEyeSimpleTransmission.BS2 Example Output**

### **RoboEye Sensor Direct Commands Control from the BASIC Stamp**

In some cases it can be very useful if the RoboEye Sensor Module could initiate a picture transmission without any user interaction from the PC.

For such an application a few "escape" sequences are defined. These start with the escape character (character number 0x1b hexadecimal, 27 decimal) and terminate with a carriage return (0x0d, or 13). The following escape commands are implemented in the RoboEye firmware:

<b>Escape T</b> Transmits a Picture	<ESC>T<CR> 0x1b 0x54 0x0d #27T#13 hexadecimal characters hidden command
<b>Escape S</b> Object detection parameters	<ESC>Sc s d<CR> c - searched color {R, G, B} s - object size {0...255} d - detected object {0, 1} 0x1b 0x53 0x31 0x30 0x30 0x20 0x31 0x30 0x20 0x0d #27SR 100 10 1#13
<b>Escape P</b> Position of the detected object	<ESC>P<CR> the picture sensor gives the x coordinate of the detected object x {0...176, 0 := no object found}
<b>Escape I</b> Command to the I2C bus at the RoboEye Sensor Module	<ESC>In m<CR> 0x1b 0x49 0x15 0x40 0x0d changes the color output sequence (blue picture) 0x1b 0x49 0x15 0x41 0x0d default value (natural picture color)

### RoboEye Sensor Module Escape Commands

The most interesting commands are the T, S and P commands. See the "Object Detection" section for more detailed use of the S and P commands.

RoboEye has several other very useful commands which can be activated from the RoboEye software or from the BASIC Stamp.

There is only one I/O line available which can be used as an input or output, but not both at the same time. To avoid data conflicts the I/O line works in half duplex mode. In output mode the UART is deactivated. Under some circumstances it would be very helpful to get more access to RoboEye Sensor Module by sending and receiving commands at the same time. For example you might want to use some of the escape commands without external hardware. For a test type "echo" on the terminal of the PC software and press enter. The sensor answers "Echo On". Now you can use the escape sequences, like "#27SR100 10 1#13". The # character signals that the following numbers specify the ASCII code of the character (#27 := 0x1b := <ESC>). The mode ends by sending "noecho".

<b>Echo On / Echo Off</b> Send and receive commands at the same time	Type "echo" in the RoboEye software terminal to enable the use of the escape sequences. Type "noecho" to turn off.
---	---

### Echo On / Off Command

Two other commands, "ping" and "color" provide the RoboEye Sensor Module firmware revision and the search parameter for the "S" command.

Two other "hidden" commands are available, "ping" and "color". The "ping" command returns the revision number of the RoboEye firmware. The "color" command is very helpful to specify the search parameter for the 'S' command.

<b>Ping</b> Obtain RoboEye firmware revision number	Type "ping" in the RoboEye software terminal to obtain the RoboEye Sensor Module firmware revision number.
<b>Color</b> Used to obtain the average color	Type "color" in the RoboEye software terminal to obtain the average color present.

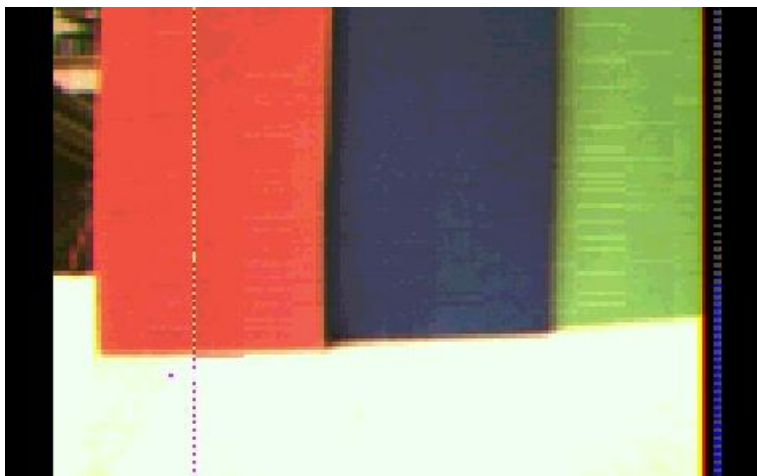
### Ping and Color Commands

### Object Detection with RoboEye

RoboEye's camera sensor is controlled by a powerful 16 bit microcontroller. Most of the processor's power is needed for data handling between the CMOS picture sensor and the wireless radio. But a portion of the program is reserved for picture processing.

The microcontroller is able to detect the x coordinates of a specified object. It helps to have a brief overview of the object detection algorithm.

The picture pixel data comes in line by line with a very high speed. Suppose you want to search out the red color.



Each color pixel consists of the color values of red, green and blue. A red pixel has a "red" value that is larger than the green and blue values. In our example the microcontroller looks at all pixel values in the line. If it finds a red pixel the x coordinate for that pixel will be stored.

But how can the RoboEye Sensor find a colored object? Objects with more than one red pixel in the line are marked so that only areas with many red pixels following each other are noted. The algorithm measures the sizes of the areas in every row, marks the largest ones and computes the main emphasis (x coordinate) of every area. The average value of all x coordinates is now the position of the object.

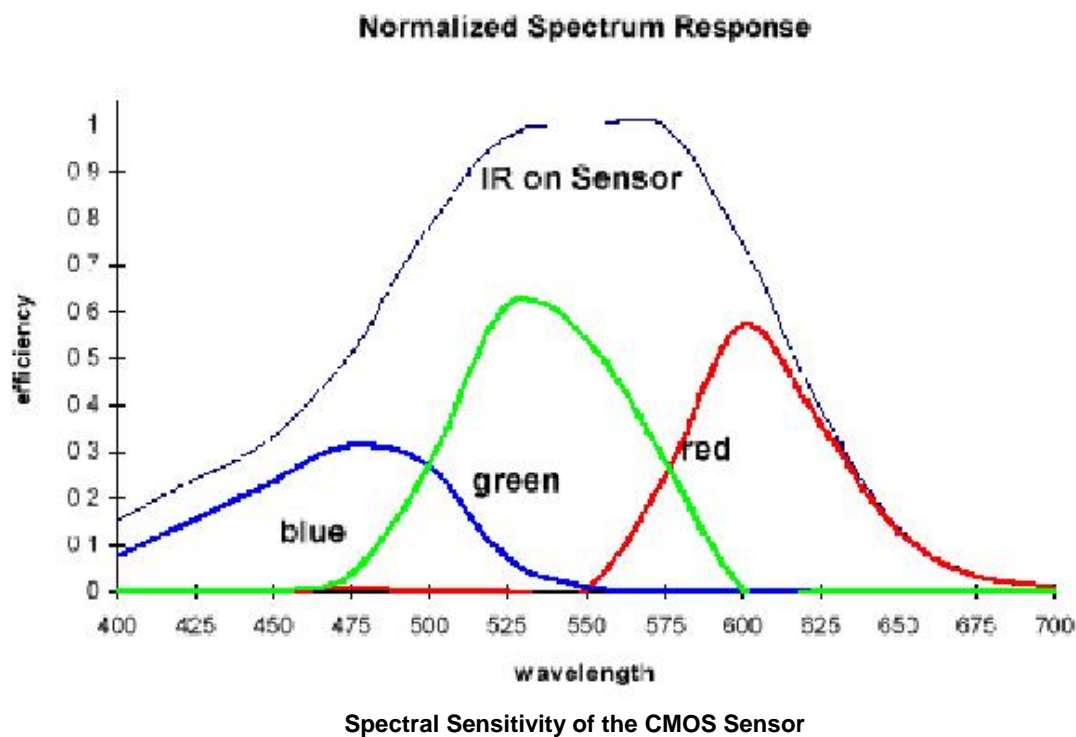
But what will happen if two nearly same sized red objects exist in different positions? Try it out.

Because of the limited computational power the search algorithm is as simple as possible. Only green, red or blue objects can be found. During development of the search algorithm it was very easy to read out the found color pixel value from the microcontroller.

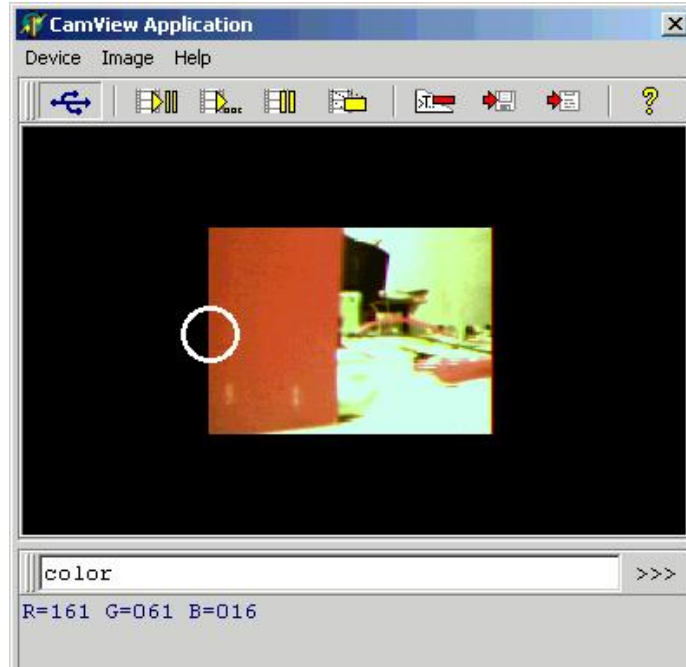
But the used CMOS sensor has its own idea of what is red. What does the sensor do? Looking into the data sheet shows how the sensor works.

Each pixel has a color filter, so that pixels with red, green and blue sensitivity exist. The sum of the three colors results in the real (natural) color of the picture detail. That looks easy, but it isn't. One of the problems is the "white balance". If you are sitting at a candle light dinner (with ordinary candles), your eyes will have a few problems separating colors.

With a CMOS sensor, an electronic system has to build the white balance, so that colors can be detected. It is also very important to know what is red, or green or blue from the view of the sensor. That is the function of the hidden command "color".



Suppose you need to know "how" red or green or blue the object is. The exact object color value gives you some information if the search algorithm detects something.



**The White Circle Specifies the Measurement Area**

See the white circle at the left of the above picture. From this area the RoboEye Sensor Module gives you the values of the red, green and blue pixel. In the example the CMOS sensor measures a value of 161 for red, 61 for green and 16 for blue.

Now you can specify your search parameter. For example you choose `<ESC>SR 80 10 1<CR>`. The value 80 means that the search algorithm looks for red pixel with a difference larger than 80 to green or blue. In other words, only pixels with  $(\text{red} > \text{green} + 80)$  and  $(\text{red} > \text{blue} + 80)$  are detected. The value 10 indicates that areas greater 10 than pixels of red in a row are noted. The number 1 at the last position tells the controller to mark found objects with a vertical line.

If you choose 120 instead of 80 no object can be found, with a value of 5 (instead of 80) the difference is too small to separate objects. In this case everything is an object, the result of the found x coordinate is useless.

The detecting algorithm is much faster than the picture transmission. The green LED on the RoboEye Sensor Module's printed circuit board flashes when no object is detected and goes on when an object is found.

## Boe-Bot Control Example

This example program demonstrates how to control the Boe-Bot using RoboEye's terminal. The program is also available for download from the RoboEye web page on [www.parallax.com](http://www.parallax.com). Once connected, press "reset" on the Board of Education. Type either F, B, L or R for direction followed by "enter" or the >>> to send a carriage return.

Enter distances in three decimals. In other words, a distance of 20 should be entered as 020 and so on.



RoboEyeBoeBot.bs2 Example Program

```

' -----[ Title ]-----
' Basic Boe-Bot Control with RoboEye - RoboEyeBoeBot.bs2
' Control the Boe-Bot with the RoboEye Terminal while observing incoming
' video feed.

' {$STAMP BS2}                                ' Stamp directive
' {$PBASIC 2.5}                                ' PBASIC directive

' -----[ Variables ]-----

pulseCount    VAR    Byte                    ' FOR...NEXT loop counter
direction      VAR    Byte                    ' Direction control
distance       VAR    Byte                    ' asdf

' -----[ Initialization ]-----

' -----[ Main Routine ]-----

controlPanel:
  SEROUT 15, 84, ["Direction?",CR]
  SERIN 15, 84, [direction]
  PAUSE 10
  SEROUT 15, 84, ["Distance?",CR]
  SERIN 15, 84, [DEC3 distance]
  IF direction = "F" THEN Forward_Pulse
  IF direction = "B" THEN Back_Up
  IF direction = "L" THEN Turn_Left
  IF direction = "R" THEN Turn_Right
  GOTO controlPanel

' -----[ Subroutines ]-----

Forward_Pulse:                                ' Send a single forward pulse.
  FOR pulseCount = 0 TO distance
    PULSOUT 12,650
    PULSOUT 13,850
    PAUSE 20
  NEXT
  RETURN

Turn_Left:                                    ' Left turn, about 90-degrees.
  FOR pulseCount = 0 TO distance
    PULSOUT 12, 650
    PULSOUT 13, 650
    PAUSE 20
  NEXT
  RETURN

Turn_Right:                                    ' Right turn, about 90-degrees.
  FOR pulseCount = 0 TO distance
    PULSOUT 12, 850
    PULSOUT 13, 850
    PAUSE 20
  NEXT
  RETURN

Back_Up:                                        ' Back up.
  FOR pulseCount = 0 TO distance
    PULSOUT 12, 850
    PULSOUT 13, 650

```

PAUSE 20  
NEXT  
RETURN

## Data Protocol Between PC and RoboEye Sensor Module

The data transmission protocol is message oriented, meaning that each payload is packed into a message before transmitting. The protocol used by RoboEye depends on the task that has been accomplished.

The wireless message is defined by the technical specifications of the radio modem. The modem works in the "shock burst" mode. In this mode 28 bytes is sent as the payload of one message. Every message is signed by an address and secured by CRC sum. Depending on the environment messages can be corrupted by other radio modems, like Bluetooth or Wireless LAN. Corrupted messages are ignored by the receiver and their payloads are lost. Corrupted messages are not repeated automatically.

The following example illustrates the structure of a message. The message contains of two header bytes and the payload. CRC and address byte are added and removed by the radio.

```

/*****
/* Structure of the wireless message protocol */
/*
/* SYNC | ID | NR | 12 pixel (26 Byte) |
/* 0xdb 0x4n 10 GR GB GR GB GR GB GR GB GR GB GR GB |
/* | |--- blue
/* | |--- green
/* |----- red
/* Color format: 4:2:2
/*
/* SYNC - 0xdb start of message
/* ID - indicates the type of the payload, i.e 0x5n == picture data
/* n (5 bit low nibble) contains the column number
/* NR - row number (0...72 for actual size)
/* Data - pixel data (12 x 2 byte = 12 pixel)
*****/
```

Because the picture sensor also transmits data from the built-in UART, other ID numbers indicate other payloads. The different numbers and the type of the messages have to be defined. The PC receives the messages and creates the picture. If some messages are lost, the referring picture part isn't refreshed. Messages from PC to picture sensor have a similar structure.

```

/*****
/* Structure of the messages to the picture sensor */
/*
/* HEADER | ID | raw data (size as given in the header) |
/* 0b.11nn.nnnn 0xkk x x x x x x ..... (size = n)
/*
/* Header - 0b.11nn.nnnn, contains SYNC (0b11) and size of the message
/* ID - indicates the type of the message
/* enum {
/* enConfigReg = 1, /* config-data for nRF2401 */
/* enDataReg /* message data */
/* };
/* xx - data
*****/
```

The transmission unit receives data with 9600 Baud. Two different data messages exist. The first type of message is a direct command to the nRF2401 radio. The payload of this message type is sent to the config register of the radio.

The other type of message is sent directly to the transmitting register of the radio. In this case the transmission unit doesn't have its own intelligence. In other words, if a user wants to make his own

programs, he has full control of the radio modem. Note that the baud rate of the transmission is fixed. Incoming data (data from unit to PC) has a baud rate of 923.076 Baud. Outgoing data, from the PC to the RoboEye Sensor, has to be sent with 9600 Baud. Because of the small power (small sized RAM) of the micro in the transmission unit a hardware handshake is implemented. After a complete message was received, the CTS signal goes high.

## Examples

### Command to USB interface

	Header	Target	Message (18 Byte)															last byte contains channel and mode							
<b>Tx-Mode</b>	211	1	142	8	28	64	224	0	0	0	0	231	0	0	0	0	231	35	239	16					
<b>Rx-Mode</b>	211	1	142	8	28	64	224	0	0	0	0	231	0	0	0	0	231	35	239	1					

### Commands to RoboEye

Adr - Address of nRF2401 (must be 231)

Cmd - Kommando an Mobilteil

- 1 - Message is sending directly to UART (variable packet size)
- 2 - Message is sending to I<sup>2</sup>C (size must be 3 byte)
- 3 - transmit command

Par - Parameter

ID - ID number (new message get a new number)

	Header	Tar.	Adr	Cmd	Par	ID	[ message size = Par								unused												]											
UART1	222	2	231	1	10	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26							
UART2	222	2	231	1	10	2	6	6	6	6	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26							
UART3	222	2	231	1	13	3	H	e	l	l	o	_	W	o	r	l	d	!	13	15	16	17	18	19	20	21	22	23	24	25	26							
IIC1	222	2	231	Cmd	ID	[ 3 Byte			unused												]																	
IIC2	222	2	231	2	1	192	18	110	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26							
				Cmd	Par	[ unused																																
SendPic	222	2	231	3	200	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26							

The USB interface works completely transparent. For transmitting data the USB unit has to be programmed as a transmitter. That means, that the programm sequence goes to the transceiver circuit. After programming the transceiver, messages can be transmitted. To receive answers from "RoboEye" the USB unit has to be programmed into receiving mode. For programming the USB interface a Baudrate of 9600 Baud must be set. The same Baudrate has to be used for transmitting messages. In receiving mode the baud rate is set to 923.076 (921600).

## Special Technical Considerations

The range of the wireless radio depends heavily on environmental conditions. In direct line of sight the range can be up to 100 feet or more. Obstructions between the transmitter and receiver, like human bodies, reduce the range. At the furthest range the error rate of the data transmission increases. This is seen when only a few percent of the picture is transmitted back to the PC.

RoboEye's data messages are secured using a fixed address, a rolling ID and CRC check. But noise at the receiver input can still produce a "correct" message with errors. The content of such a message might include characters like "5fdg#~^" or others you did not expect.

Avoid touching RoboEye's electronic components and antenna. Electrostatic charges can damage the electronic components.

## Additional Resources

OVT Web Site

---