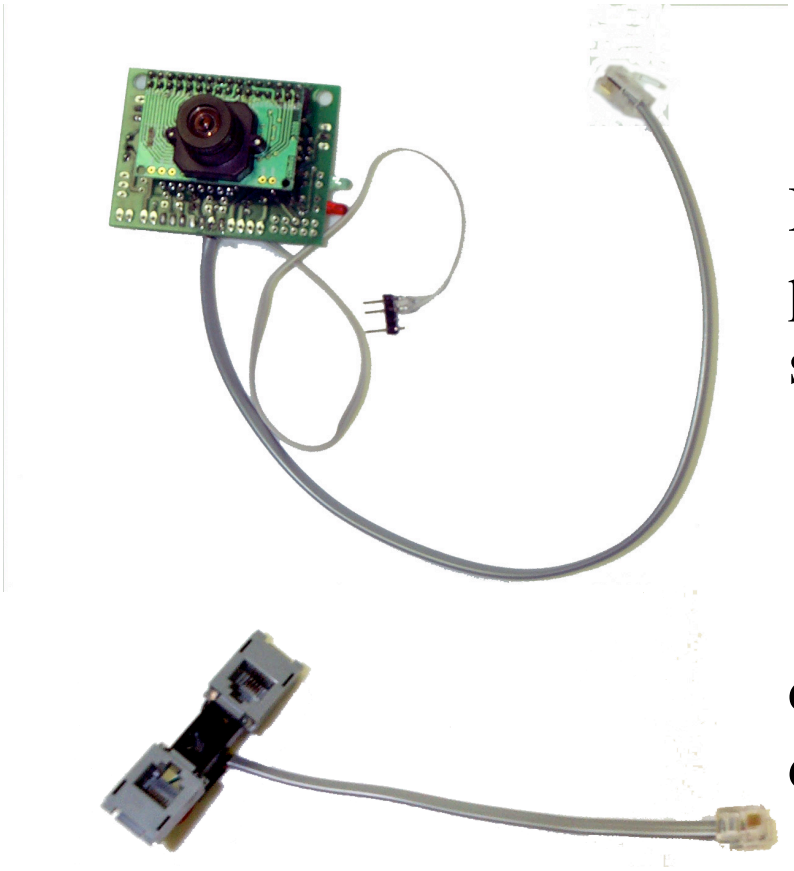
# This Year's Goody Bag

# Color Supplement

- The Goody Bag contains:
  - a CMUcam color tracking camera
  - HB-cam interface dongle
- IC 4.2 contains a new library **`cmucamlib`** which has routines to track the position and size of colored objects in the camera image
- CMUcamGUI java software which can be used for displaying images and testing tracking

# Goody Bag Contents



1 CMUcam plugs into red socket in HB-cam dongle; 3-pin connector plugs into any HB analog or digital sensor port

1 HB-cam dongle: plugs into HB; cable from interface board plugs into dongle

Botball

# The HB-cam Dongle

- The dongle allows you to charge and Zap charge your HB while the camera is connected
- The dongle has a 3-position switch:
  1. Switch by red dot (camera connector): HB is talking with the camera
  2. Switch in middle: CMUcam talks with host PC
  3. Switch away from red dot (interface board connector): HB talks with host PC
- The HB continues to charge no matter what the switch position

# Charging the HB With the CMUcam Connected

- The CMUcam uses as much power as the HB
- Together they use more power than is supplied by normal charge
- When the camera is connected and the HB is turned on, it is useful to have the charger in ZAP mode. The charger may be left in ZAP for a few minutes, even if the HB is turned off.
- **ALWAYS reset the charger to normal mode and turn off the HB when you leave the HB unattended!**

# `cmucamlib` Routines

- To use any camera routines, be sure to put `#use "cmucamlib.ic"` at the top of your file
- Call `init_camera();` to initialize camera before any other camera calls
- Use `clamp_camera_yuv();` to automatically set camera for the current lighting conditions. Camera should be pointed at a white surface when this call is being made (it waits for start button to be pressed). It takes 15 seconds for this function to complete!

# `cmucamlib` Routines (cont.)

- Call `track_blue();` & `track_orange();` to check for color blobs that CMUcam can see. These functions return 0 if they find no color blob, or the confidence of the blob detected. A good confidence is 80 and up. A confidence of 4 or 5 is poor.

# cmucamlib Routines (cont.)

- The track_color information is stored in globals:
  - `track_size` stores the approximate number of pixels matching in the blob
  - `track_x` stores the pixel x coordinate of the centroid of the color blob
  - `track_y` stores the pixel y coordinate of the color blob (note: 0,0 is the center; 40,80 is upper right and -40,-80 is lower left)
  - `track_area` stores the size of the bounding rectangle of the color blob
  - `track_confidence` stores the confidence for seeing the blob

# cmucamlib Routines (cont.)

- For experts: use `trackRaw(...);` to specify a particular color for tracking. Returns 0 if no such blob is found, -1 is there is a communication error, or the confidence. Also check out `setWin(...);`
- More details and low level functions are given in the comments at the beginning of **`cmucamlib.ic`** and **`cmucam3.ic`** in your Handy Board lib folder

# cmucamlib-demo.ic Example

```
/* demonstrate color blob sensing for poof balls and blue paper */
#use "cmucamlib.ic"
void main()
{
    init_camera(); // initialize the camera in YUV mode
    clamp_camera_yuv(); //clamp camera white balance in YUV mode
    while(!stop_button()) { // hold down Stop for a long time
        if (track_blue() > 4) { // you could make this 0 bigger
                                // number, like 80 for example
            printf("blue found:%d\n", track_confidence);
        } else
          if (track_orange() > 4) {
              printf("orange found:%d\n", track_confidence);
          } else {
              beep();
              printf("nothing...\n");
          }
    } // end while()
} // end main()
```

# `cmucamlib` Notes

- Don't forget the `#use "cmucamlib.ic"`
- A handyboard that has been loaded with cmucamlib can only have a new program downloaded by **turning on the power while the start button is pressed**
- More details and low level functions are given in the comments at the beginning of **`cmucamlib.ic`** and **`cmucam3.ic`** in your Handy Board lib folder

# Critical Things to Know if You Don't Want to Embarrass Yourself at the Tournament

# Utilities for Botball

- The library file *botball.ic* contains special routines to help you write your Botball programs
- the function `wait_for_light(`<*port*>`)` runs the calibration routine for the Handy Board or RCX with the light sensor plugged into <*port*> and then waits for the light to come on
- When using this function write:

```
#use "botball.ic"
```

at the top of your program file (before main)

Botball

# Timing for Botball

- When executed, the function

  `shut_down_in(`<*game_secs*>`);`

  starts a process that turns off all motors after *game_secs* has elapsed and keeps any new commands from being processed until the Handy Board or RCX is power cycled (servo motors, if running, are actually "neutralized" in their last position rather than turned off)

Note: *game_secs* must be a float number (with a decimal point)

# Template for Tournament Code

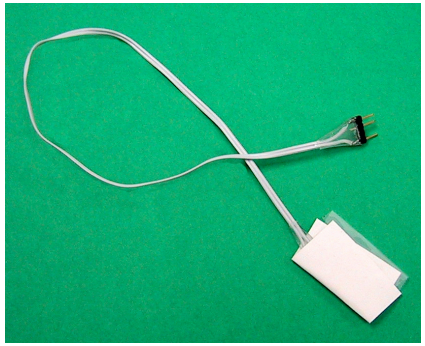
```
/* load the Botball utilities */
#use "botball.ic"
/* Botball.ic includes the functions: wait_for_light(int port)
   and shut_down_in(float seconds)
*/
void main()
{
    wait_for_light(2);     /* light sensor in port 2 */
    shut_down_in(89.9);    /* kill the robot after 89.9 seconds */
    my_botball_program(); // call the function that does everything
}

void my_botball_program()
{
    /* my botball robot just beeps every 3 seconds */
    while(stop_button()==0){
       beep();
       sleep(3.0);
    }
}
```

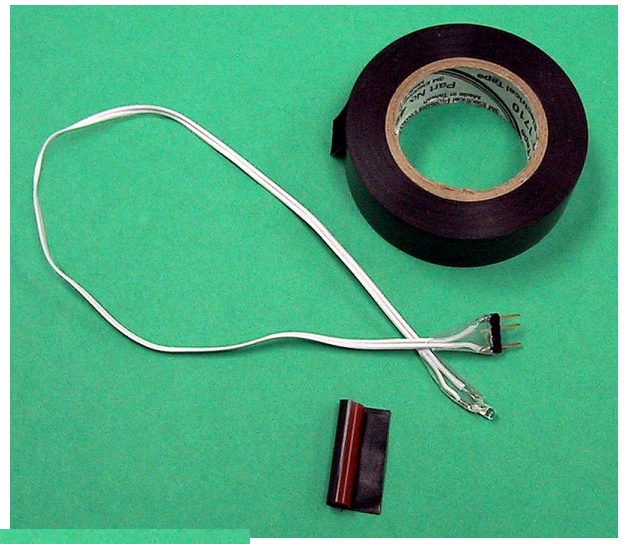
# YOU MUST SHIELD YOUR LIGHT SENSOR

- The 400 watts of halogen overhead lights from the game table will flood an unshielded sensor rendering it incapable of seeing the starting light
- Sensors only need a little light to work, and it should be shielded from all extraneous sources
- Opaque objects stop light (e.g., foil, black electrical tape)
- Soda straws are not opaque; Printer paper is not opaque; Two layers of printer paper are not opaque; A straw wrapped in printer paper is not opaque.

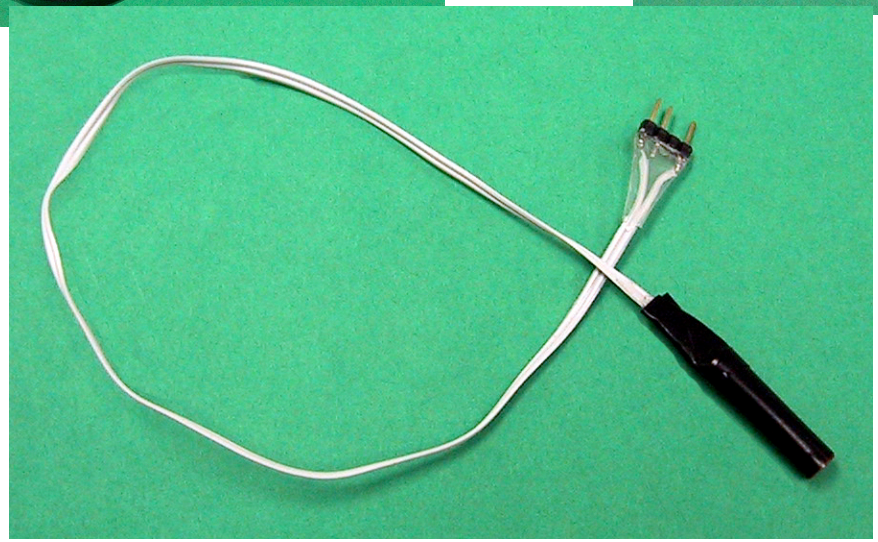
# Shielded Sensors

No!!





Yes!



Botball