

March 01, 2008

## Lab 9: Color Sorting of Objects

Bernard M. Gordon Center for Subsurface Sensing and Imaging Systems (Gordon-CenSSIS)

---

### Recommended Citation

Bernard M. Gordon Center for Subsurface Sensing and Imaging Systems (Gordon-CenSSIS), "Lab 9: Color Sorting of Objects" (2008). *High Tech Tools & Toys Labwork*. Paper 9. <http://hdl.handle.net/2047/d20003902>

This work is available open access, hosted by Northeastern University.

## GEU111 – High-Tech Tools and Toys Lab

### Lab 9: Color Sorting of Objects

In this lab, we will apply what we have learned about color identification and stepper motor control to create a system to sort ping-pong balls on the basis of color. Figure 1 below shows the ball sorting apparatus. A transparent tube holds the input balls and a stepper-motor-controlled rotating receptacle holder is positioned under the input column. A video-cam mounted on the ring stand that holds the input column is focused on the lowest ball. Based on the color of the lowest ball, the stepper motor rotates the appropriate receptacle under the input column and releases the ball to let it fall into the right receptacle.

The lab will be done in steps: first you will write a program that sorts ping-pong balls of two colors, one at a time. Second, you will write a program that works with a stack of balls. The program will identify the color of the lowest ball, decide which receptacle the ball should go into, rotate the appropriate receptacle under the column of input balls, release only one ball, and repeat until the input column of balls is empty.

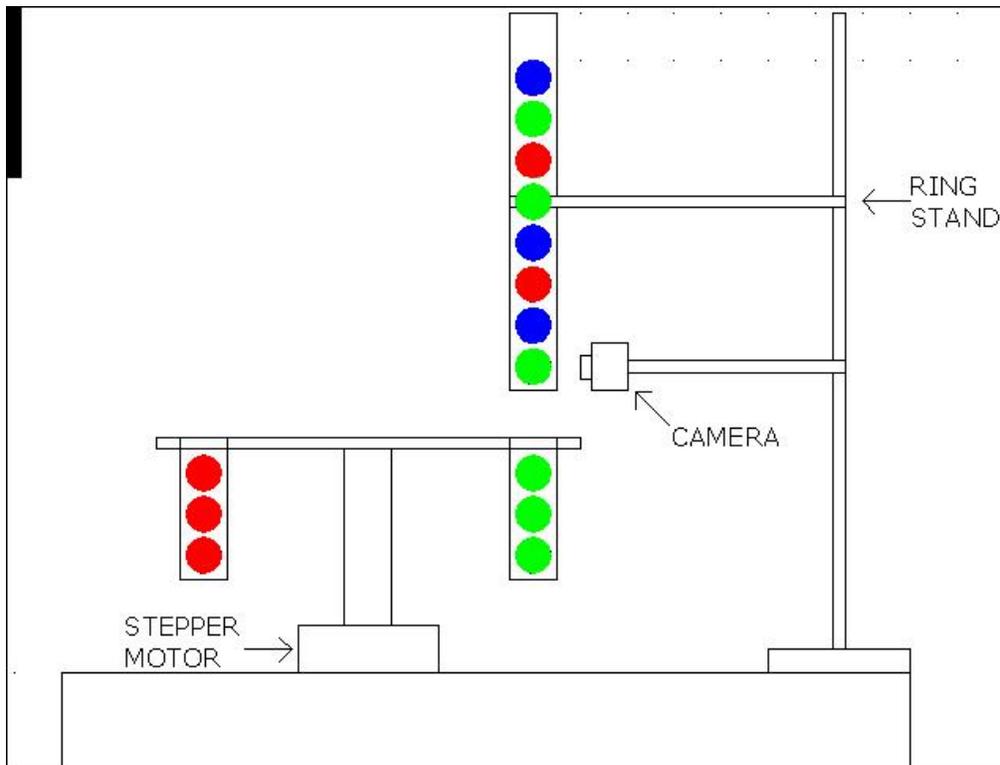


Figure 1: Object color-sorting apparatus.

## A. Operator-Assisted Single Ball Sorting System

Position the input column on an arc between receptacles 1 and 2, close enough to the turntable that a ball cannot get out except when a receptacle is rotated underneath. Position the video-cam so that a ball at the bottom of the column will be in the center of the field of view. Write a program that does the following:

- i. Prompts the user to drop a red or a green ball into the input tube.
- ii. Captures the image and downloads the picture to an array
- iii. Converts the array into a matrix
- iv. Breaks each pixel into the R, G, and B values and defines red, green, and blue matrices
- v. Averages the r, g, and b values of the pixels that correspond to the ping pong ball. (You can determine beforehand how much and what part of the image corresponds to the target ball.)
- vi. Converts the average r-g-b value of the ball into a h-s-v (hue-saturation-value) triple.
- vii. Rotates the #1 receptacle underneath the input column if the ball is red or rotates the #2 receptacle underneath if the ball is green.
- viii. Rotate the receptacle holder back to half-way between receptacle #1 and #2 and prompt the user to put another red or green ball in the input tube.
- ix. If the ball is neither red nor green, the program should display: “Ball is not red or green” and wait for operator assistance.
- x. Your program should work regardless of the level of illuminator on the balls

In this program you will need to position the input column an integral number of steps between the two receptacles. You can just count steps to move receptacle #1 or 2 under the input column and then back to the midway point. A more sophisticated control system would use the photosensors on the rotary dial to tell you when the proper receptacle is underneath the column. For an Extension problem, modify your program to use the analog input functions `AI_Configure()` and `AI_VRead()` to determine when the right receptacle is under the input column.

As described, your program will only be able to sort two colors of ball. Experiment to see if you can move the holder fast enough to skip over a receptacle and onto the next receptacle. If so, you can add code to sort blue balls into receptacle #3 and white or any other color into receptacle #4. Can you make your program take more than one ball in the input column for sorting?

## B. Automatic Ball Sorting System

To drop just one ball from a column of balls, you will probably need something to hold up the balls until the input column is over the correct receptacle, release just one ball, and then prevent the balls from dropping again. A *linear actuator* is an electro-mechanical device that converts an electrical current into a linear motion of a shaft. Most inexpensive linear actuators use a switched electrical current from a controlled power

source to drive an electromagnet. The electromagnet then attracts a magnetic shaft to move into the electromagnet solenoid.

So all you need to do is to turn a voltage on or off to get a linear motion of a shaft. Unfortunately, it takes more current to turn on the electromagnet than you can get out of the digital outputs that we have been using to turn on LED's or control the stepper motor. "TTL outputs" (short for Transistor-Transistor Logic) such as the bits in the Digital Output bus can source or sink less than about 20 mA of current, while the electromagnet in the linear actuator requires 100's of mA of current.

There are two ways to address this problem. The DC power supply on your lab bench can output up to 1A of current at 12V. You can set the power supply at 12V to drive the linear actuator and then use an external *transistor* to switch the power with a digital output port, using `DIG_Port_Out()` as we did in Lab 7 when we turned LED's on and off while controlling the stepper motor. A transistor is a device that uses a small amount of current to control a larger current – you can think of it as opening or closing a switch. The circuit is shown at the end of the lab for those of you who are interested. This kind of circuit is commonly used to switch large currents from a computer output.

The second way you can switch a large current with your computer program is to control the power supply directly through the IEEE-488 bus that connects all the instruments on your bench to a controller card in the computer backplane. The IEEE-488 bus is also called the "Hewlett-Packard Instrument Bus" (HP-IB) by the Hewlett-Packard Company (now Agilent), a leading manufacturer of computer-controlled test equipment, such as that in the High-Tech Tools and Toys Lab. HP-IB is called the "General Purpose Interface Bus" (GPIB) by other manufacturer.

The software that controls the GPIB/HP-IB controller card is called the Standard Instrument Control Library (SICL). It consists of a series of C/C++ routines that send text strings to the appropriate instruments to control all the functions of the instruments that can be accessed from the front panel buttons.

To use the SICL routines you need to:

a) `#include` a header file "`sicl.h`" in your program file

b) add the library file "`sicl.lib`" to your project

[ "`sicl.h`" and "`sicl.lib`" are stored in the National Instruments\NI-DAQ\Include directory on the computers in the HTT&TL ]

c) include a preprocessor "`#define`" command (like your `#include` statements) to define the HP/Agilent 3632A (or HP/Agilent 3631A) as instrument 5 on the GPIB bus:

```
#define POWERSUPPLY "gpib0,5" // HP E3632A is GPIB0::5::INSTR
```

In the body of your program you need the following statements:

```
INST id; // defines id as a "device session" variable
```

```

id = iopen(POWERSUPPLY); // Open a device session using define address

iprintf(id, "*RST\n"); // Write the *RST string to put the
instrument in a known state.

iprintf(id, "VOLT 12.0\n"); // set power supply output to 12V
iprintf(id, "OUTPUT:STATE ON\n"); // enable power on
Sleep(500); // set delay to let ball drop
iprintf(id, "OUTPUT:STATE OFF\n"); // enable power off

fclose(id); // close the session before exiting

```

When you have decided how you want to control the linear actuator and determined that you can make it work, connect the linear actuator to the input ball column using the clamp provided so that the linear actuator projects through the hole near the bottom of the column. When the power is turned on the linear actuator will retract and the ball will fall. You will have to test out the timing to see how long you need to turn the power supply (or relay) on to let one ball, and only one ball, drop out of the input column. This may vary with the number of balls in the column. If necessary you may have to change your delay based on the number of balls left in the column.

Write a program that does the following steps after you load the input column up with mixed colors of balls:

- i. Prompts the user to start the sort and asks how many balls are in the input
- ii. Captures the image of the lowest ball and downloads the picture to an array
- iii. Converts the array into a matrix
- iv. Breaks each pixel into the R, G, and B values and defines red, green, and blue matrices
- v. Averages the r, g, and b values of the pixels that correspond to the lowest ping pong ball. (You can determine beforehand how much and what part of the image corresponds to the target ball.)
- vi. Converts the average r-g-b value of the ball into a h-s-v (hue-saturation-value) triple.
- vii. Rotates the #1 receptacle underneath the input column if the ball is red, the #2 receptacle underneath if the ball is green, the #3 receptacle underneath if the ball is blue, and the #4 receptacle underneath if the ball is white or any other color.
- viii. Activates the linear actuator for enough time to let the ball drop.
- ix. If the next ball is the same color, operate the actuator to drop it into the same receptacle. Otherwise move the proper receptacle under the input column.
- ix. Repeat the process until all the balls are sorted.

## Extensions

1. Use the analog input functions to read the photosensor inputs and write a control program that will drop the red, green, and blue balls in the correct receptacles (#1, #2, and #3 respectively) no matter where the stepper motor is located when your program starts. Your program will have to determine where it is located at the beginning of the experiment by sensing the output from the photoresistors.

2. Getting the timing of the linear actuator right so you drop only one ball may be tricky. If you had a second linear actuator above the bottom ball as well as the actuator below the bottom ball, you could release just one ball at a time. Design a program to do the multi-ball sorting with two linear actuators controlled with two digital outputs (DIO2 and DIO3). Note that in this case, you can't just control the power supply voltage over the GPIB bus because you will need to independently control two linear actuators. You will need to fix the power supply voltage at 12 volts and switch the current on and off with transistors controlled by the digital outputs as in the circuit below:

