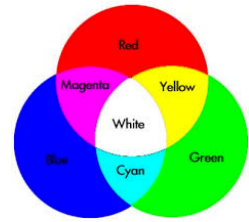


Learning Matrix Equations through Color Mixing

Prof. John Wen, TA: Zhaoyi Zhang, Matt Minakais



Objective

This project illustrates an application of matrix equations through color mixing. Students will learn to use matrix equation solution to find LED voltage inputs to achieve the desired output color.



Experiment

The light mixing apparatus consists of six LEDs, a mixing tube, and an RGB color sensor. The LEDs and the sensor are interfaced to the National Instruments MyRIO. The voltages of the LEDs are adjusted to produce different colors, which are measured by the color sensor. The students will first find the light propagation matrix relating the input voltage to the sensor output. This matrix is then used to find the input voltage generate any desired color. As the system has more inputs (6) than outputs (3) and the inputs are constrained to be positive with upperbounds, non-uniqueness of solution and optimization-based solution will also be considered.

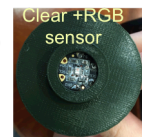
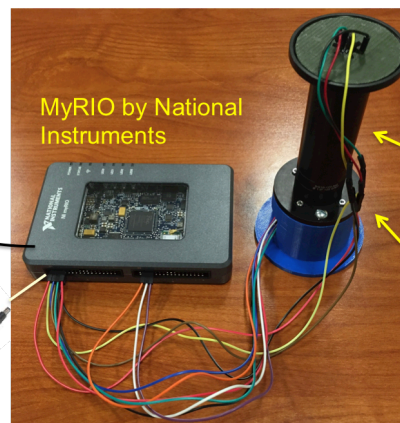
Parts

- Six LEDs mounted on a custom board
- Color sensor
- Color mixing tube
- MyRIO

Windows 7 or 8.1 PC



USB



Background Reading

- Color science:

<http://www.cs.cornell.edu/courses/cs465/2005fa/lectures/23color/23color.pdf>
<http://docs-hoffmann.de/ciexyz29082000.pdf>

- LEDs and color sensors: npoints.com: [LEDs](#) , [Color Sensor](#)
- PWM (Pulse Width Modulation) is used to vary the current to the LED to control its luminosity. With 100% duty cycle, the PWM pins of myRIO will output 5V. The current applied to the LED is calculated by: Duty Cycle (in percentage) * Voltage Output (5V for myRIO) / resistance.
More reading: <https://www.arduino.cc/en/Tutorial/PWM>

Software Installation

System Requirement: NI MyRIO is compatible with Windows 8.1/8/7/Vista/XP/Server

Installation:

Two discs are provided in software bundle of NI myRIO kit. Insert DVD1 and install default selection. To activate modules, serial number "S/N" can be located on a label on the front of software bundle package. <http://www.ni.com/myrio/setup/getting-started/>

Program Setup:

Download the project software to a directory. Click on "LED RGB Mixing" LabVIEW Project file (LED RGB Mixing.lvproj). On the LabVIEW window, click "Launch LabVIEW". In the "Project Explorer" window, expand "Project:LED RGB Mixing.lvproj" and expand "myRIO-1900(...)". Double click "LED Power Sliders.vi" and the front panel will be open. Before executing the program, make sure myRIO is connected to power and the computer. Click on a right arrow, the run icon, on the top left and execute the program. <http://www.ni.com/product-documentation/14603/en/>

Setup

The kit consists of two boards: a LED control board and a sensor board, both of which needs to be connected to MyRIO port.



MyRIO Port A			
Pin 33 3.3V	Pin 31 Blue	Pin 29 Green	Pin 27 Red
Pin 34 SDA	Pin 32 SCL	Pin 30 GND	Pin28 GND

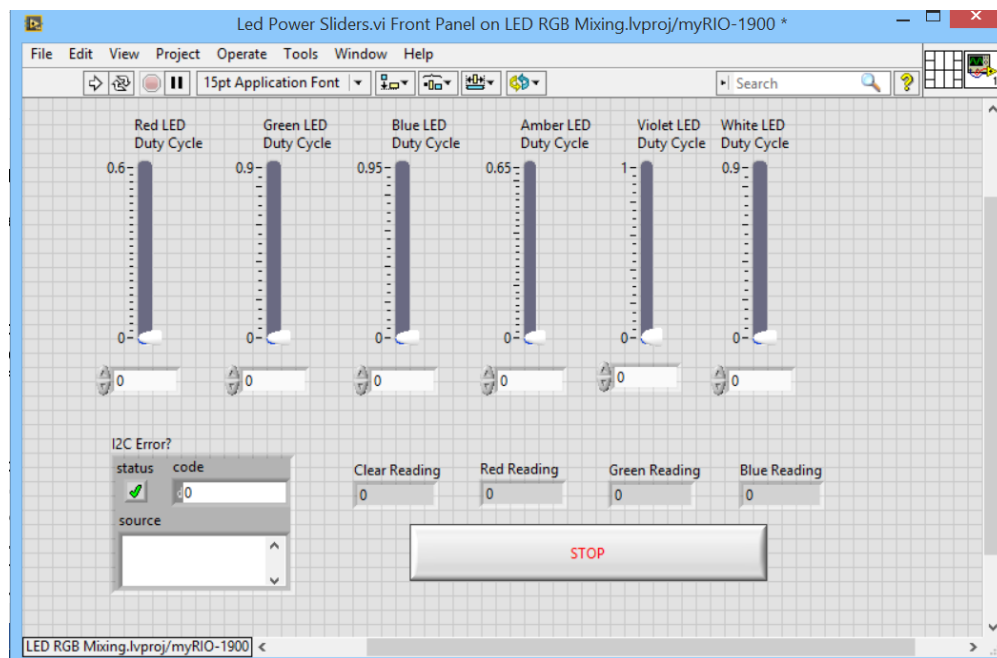
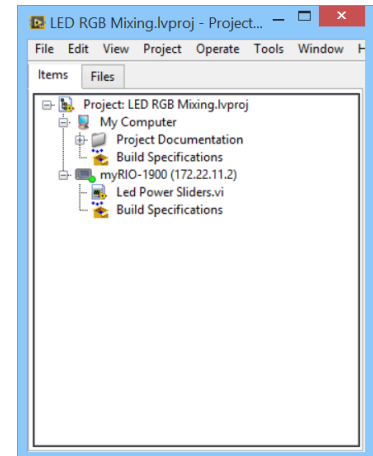
MyRIO Port B			
	Pin 31 Amber	Pin 29 White	Pin 27 Purple

LED Control Board		Sensor Board	
Red LED	Pin 27/Port A	3.3V	Pin 33/Port A
Green LED	Pin 29/Port A	GND	Pin 30/Port A
Blue LED	Pin 31/Port A	SDA	Pin 34/Port A
Amber LED	Pin 31/Port B	SCL	Pin 32/Port A
White LED	Pin 29/Port B		
Purple LED	Pin 27/Port B		
GND	Pin 28/Port A		

Procedure

Starting the experiment

1. Connect the wires from the experiment to the MyRIO as discussed above.
2. Connect power to MyRIO and plug MyRIO into USB port of the laptop.
3. Start the LabView program.
4. Choose LED Power Sliders.vi. A window will pop up as shown.
5. In the window, there are six slider dials to adjust voltage into each of the six LEDs, and the sensor read out from RGB and clear sensors.
6. Press  to start the program. Press  to stop the program.
7. Try adjusting the slider or type in some values for the LED inputs, and check the sensor values and the actual colors.



Exercise 1: Find the matrix mapping the input to output.

There are six inputs and four outputs.

The inputs are the duty cycle of the 5V into each LED, red R, green G, blue B, Amber A, Violet V, White W. We denote them by $(u_1, u_2, u_3, u_4, u_5, u_6)$. Each LED has a maximum

allowable current, so the input is constrained by the maximum duty cycle: $0 \leq u_i \leq u_{i_{\max}}$ where u_{\max} is given by (0.6,0.9,0.95,0.65,1.0,0.9).

The outputs measure intensity of light in clear (unfiltered), red, green, blue channels. Denote the output measurements by (y_1, y_2, y_3, y_4) corresponding to the clear, red, green, and blue light intensity.

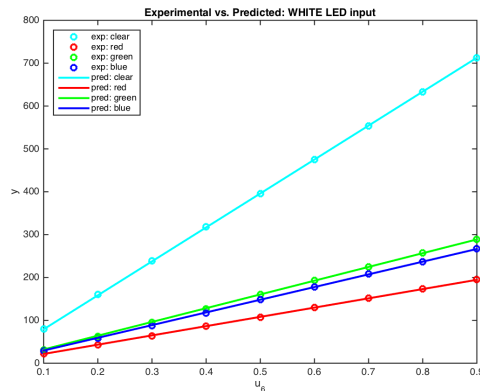
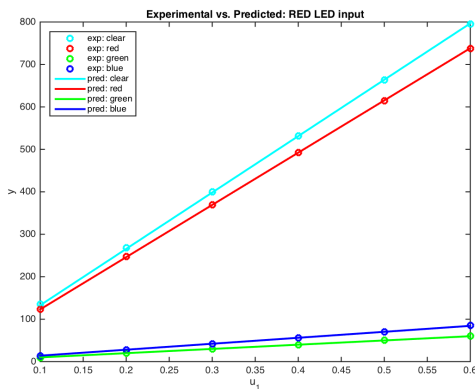
The relationship between the input voltage and output light intensity is almost linear, so we model the input/output relationship by a 4x6 matrix:

$$y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} & A_{15} & A_{16} \\ A_{21} & A_{22} & A_{23} & A_{24} & A_{25} & A_{26} \\ A_{31} & A_{32} & A_{33} & A_{34} & A_{35} & A_{36} \\ A_{41} & A_{42} & A_{43} & A_{44} & A_{45} & A_{46} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix} = Au$$

1. Choose a set of input voltages u (type them in the box corresponding to each LED) and record the corresponding sensor output y in a spreadsheet.
2. Use these data to find the matrix A .
3. Compare the experimental output values y corresponding to u with your model predicted output Au .

Your answer should include

- a. Matrix A that you have obtained
- b. Inputs that you have used, the outputs that you have measured, and the predicted output using A .
- c. Plots comparing experimental (measured) output and predicted output (examples shown below).



For Part 2, there are two common methods:

Method 1: Step through different values for each input (e.g., set u_1 to 0.1, 0.2, etc., and all other u_i 's to zero, etc.), and record the corresponding output y . Let's say you have N total input/output pairs. Put all the inputs together into a $6 \times N$ matrix U and all the outputs together as a $4 \times N$ matrix Y . Then our model predicts $Y = AU$. To estimate A , use the MATLAB command $A = Y \cdot \text{pinv}(U)$ (see if you can figure out how to do this in LabView – search online!). The command pinv denotes pseudo-inverse (you will learn more about this important concept in later courses). In the case that a matrix is square and invertible pinv is the same as inv . But pinv works for any matrix.

Method 2: This is similar to Method 1, but much faster. Use uniform random generator to generate a random input vector u (with the amplitude constrained between 0 and u_{\max}). Do this for N random inputs (see if you can determine a good number – shouldn't need to be too large) and record the corresponding outputs. Form U and Y and then solve for A as in Method 1.

Optional: Program LabView to automatically find A for you, so you can just press a button, then the inputs and outputs are automatically generated and acquired to find the estimated A matrix.

Exercise 2: *Use RGB LED to generate specified colors.*

This exercise will use only the RGB LEDs to achieve the desired RGB sensor outputs.

Let the matrix B be the submatrix of A that corresponds to RGB inputs and outputs (i.e., rows 2-4 and columns 1-3). So B is a 3×3 square matrix.

1. Use the inverse of B to find the RGB LED inputs to achieve the following colors (the values shown are the corresponding RGB intensity values):
 - (i) blue sky (135,206,250)
 - (ii) sunset (229,123,110)
 - (iii) turquoise pond (158,215,194)
2. Use an online color picker tool, e.g., rapidtables.com/web/color/RGB_color or colorpicker.com to compare by eye the colors specified above with the colors from the LEDs. To see the color generated by the LEDs, remove the sensor from the color mixing tube, put a thin sheet of white paper (e.g., tissue paper) over it to see the mixed color.

Your answer should include:

- a. For 1(i)-1(iii), record the input u , the measured output y , and the desired y . Explain why the measured y and the desired y are not exactly the same.
- b. Explain possible sources of discrepancy between the specified colors and your observed colors.

Exercise 3: Use all Six LEDs to generate specified colors.

This exercise will repeat Exercise 2 using all 6 LEDs.

When there are more inputs than outputs, there are infinitely many solutions. One possible choice of the solution is to square down the system by limiting the number of input degrees of freedom: $\mathbf{u} = \mathbf{A}^T \mathbf{v}$ where \mathbf{v} is a 3x1 vector. Then $\mathbf{y} = \mathbf{A}\mathbf{u} = \mathbf{A}\mathbf{A}^T \mathbf{v}$ and $\mathbf{A}\mathbf{A}^T$ is a 3x3 matrix. Solve for \mathbf{v} , and then compute $\mathbf{u} = \mathbf{A}^T \mathbf{v} = \mathbf{A}^T (\mathbf{A}\mathbf{A}^T)^{-1} \mathbf{y}$. $\mathbf{A}^T (\mathbf{A}\mathbf{A}^T)^{-1}$ is called the right inverse of \mathbf{A} , since $\mathbf{A} [\mathbf{A}^T (\mathbf{A}\mathbf{A}^T)^{-1}] = \mathbf{I}$, the 3x3 identity matrix. Recall pinv (pseudo-inverse) from Exercise 1: $\text{pinv}(\mathbf{A})$ is the same as $\mathbf{A}^T (\mathbf{A}\mathbf{A}^T)^{-1}$. This solution is also called the "least-square" solution, because it finds \mathbf{u} that minimizes the size of \mathbf{u} : $\mathbf{u}^T \mathbf{u}$.

1. Repeat part 1 in Exercise 2 using $\mathbf{A}^T (\mathbf{A}\mathbf{A}^T)^{-1}$ to find the input \mathbf{u} .

Your answer should include:

- a. For the three colors in Exercise 2, record the input \mathbf{u} , the measured output \mathbf{y} , and the desired \mathbf{y} . Compare your solutions with the solutions in Exercise 2. Explain any difference that you have observed.

Exercise 4: Solve color mixing as an optimization problem.

For a given \mathbf{y}_{des} , we can write the problem of finding the input \mathbf{u} as a minimization problem: $\min_{\mathbf{u}} J(\mathbf{u})$, where $J(\mathbf{u}) = (\mathbf{A}\mathbf{u} - \mathbf{y}_{des})^T (\mathbf{A}\mathbf{u} - \mathbf{y}_{des}) = (\mathbf{u}^T \mathbf{A}^T \mathbf{A} \mathbf{u} - 2\mathbf{u}^T \mathbf{A}^T \mathbf{y}_{des} + \mathbf{y}_{des}^T \mathbf{y}_{des})$. Make a small change in \mathbf{u} , then $\Delta J = 2 (\mathbf{A}\mathbf{u} - \mathbf{y}_{des})^T \mathbf{A} \Delta \mathbf{u}$, choose

$$\Delta \mathbf{u} = -k \mathbf{A}^T (\mathbf{A}\mathbf{A}^T)^{-1} (\mathbf{A}\mathbf{u} - \mathbf{y}_{des})$$

Use $k=1$; this is called the Newton-Raphson method. If you start initially with $\mathbf{u}=0$ and $\mathbf{y}=0$, then the first iteration is exactly the right inverse solution.

1. In Exercise 3, the output \mathbf{y} will have some error from your desired \mathbf{y} . Use the update method above to update the input \mathbf{u} until the output is within 1 unit of the desired output.

Exercise 5: Use Quadratic Programming to deal with infeasible input

For some outputs, the pseudo-inverse solutions contain negative or large inputs which are infeasible. We can use quadratic programming, which is just the minimization problem in Exercise 4 but with the inequality constraints on \mathbf{u} , to solve for feasible \mathbf{u}

while minimizing the output error. (Look up quadprog in MATLAB or find equivalent function in LabView.)

1 Specify the output as $RGB=[300, 200, 100]$. Some of the inputs are negative. Try two solutions:

(i) Set the negative inputs to 0.

(ii) Use quadratic programming to find a feasible solution

a. Compare the inputs and output error using the two methods above.