MEMO

Drexel University

To: Dr. Christopher Peters, ECE 303 Fall 2020

From: Dinh Nguyen

Date: Oct 26th, 2020

Re: Lab 5 Graphical User Interfaces

Purpose

The goal of this lab is to continue the study of lab 3 and 4 on the photoresistor and the relationship between the photoresistor and the LED. The lab also covers basics about serial communication, how Arduino communicates through serial terminal to another local scripts (Python or MATLAB). From that, in lab 5, we will use the data from the Arduino via serial communication to establish an user interface in Python or MATLAB script.

Discussion

Hardware: Figure 1 shows how the photoresistor and the LED are connected in series with a 1k Ohm resistor to the Arduino board. Photoresistor will be connected to pin A0 to read its voltage in bits. LED will be connected to pin 5 to be controlled by analogWrite and duty cycle to adjust its brightness. In this lab, only RED LED is used for the experiment.

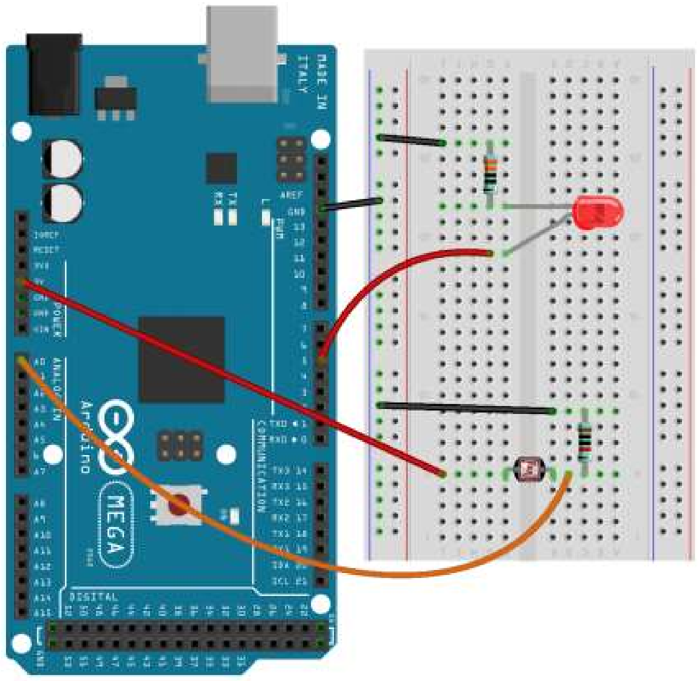


Figure 1. Photoresistor and LED circuit

Software: In order to change the duty cycle over time, I uses analogWrite and the duty cycle will be incremented by 1% which is 2.55 bits at a time. There will be 5 reading at each iteration so retry variable will keep track of that. We will have 100 iteration in total matching 100% duty cycle. The duty cycle will be reset when reaching 100%. Using analogRead, photocell reading will be recorded in bits and send to the Python script and from that, the Python script will read data from the serial, check if the data is valid (within +- 1 standard deviation) and then calculate the average voltage from 5 points of each iteration. Using the results, we can calculate the voltage across the photocell by dividing the reading by 255 bits and multiply by 5. Since the photocell is in series with the resistor, the voltage across the resistor in photocell circuit is 5 – voltage of photocell. On the other hands, the voltage across the LED is equivalent to 5 (the input voltage) multiplied by the duty cycle. Thus, the voltage across the resistor of the LED circuit is 5-voltage of LED. We will write the data collected to a log file and use matplotlib to develop some plots to study the relationship between LED and photocell.

For the user interfaces, I use the tkinter library in Python to set up a start button and a duty cycle information. The script will start when the user clicks on the start button and the serial communication will start between the Python script and Arduino. All the data will be plotted into 5 graphs as in figure 2.

To run, upload the Arduino sketch to the board and run python script by command ‘python3 lab5.py’.

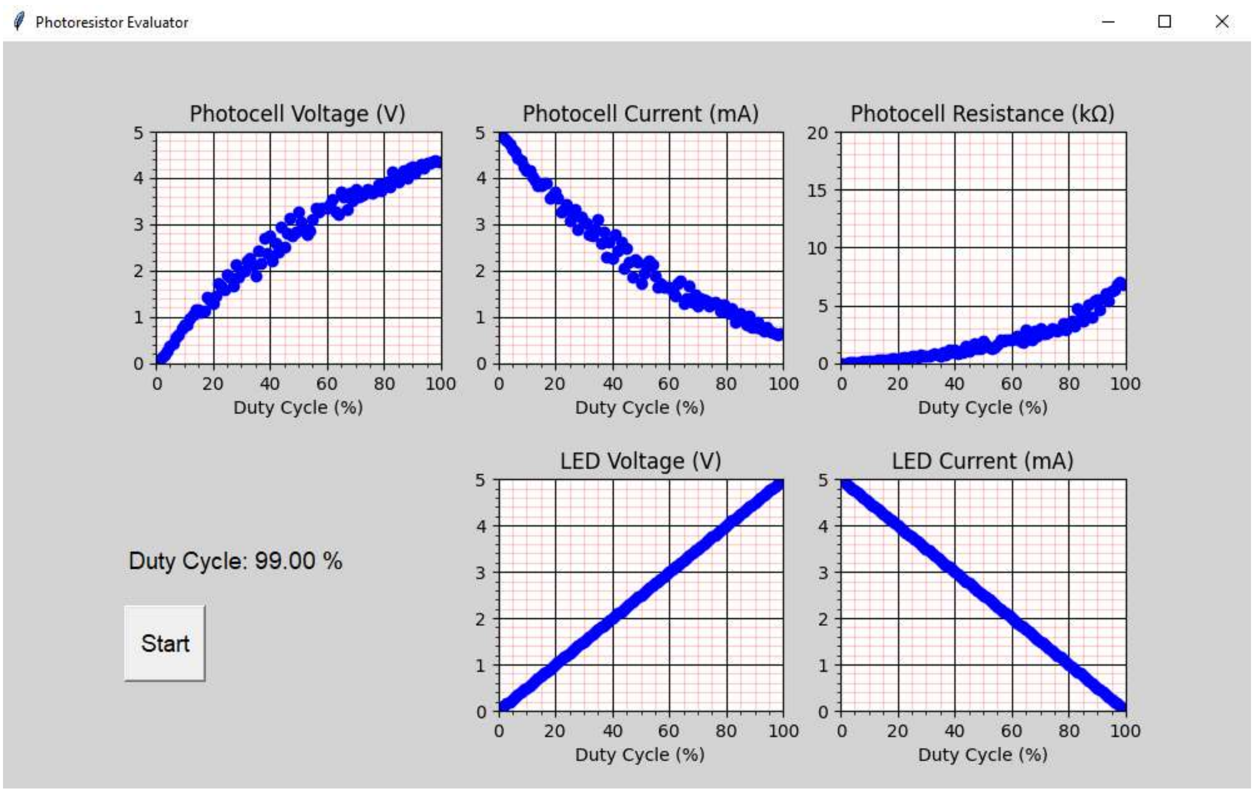
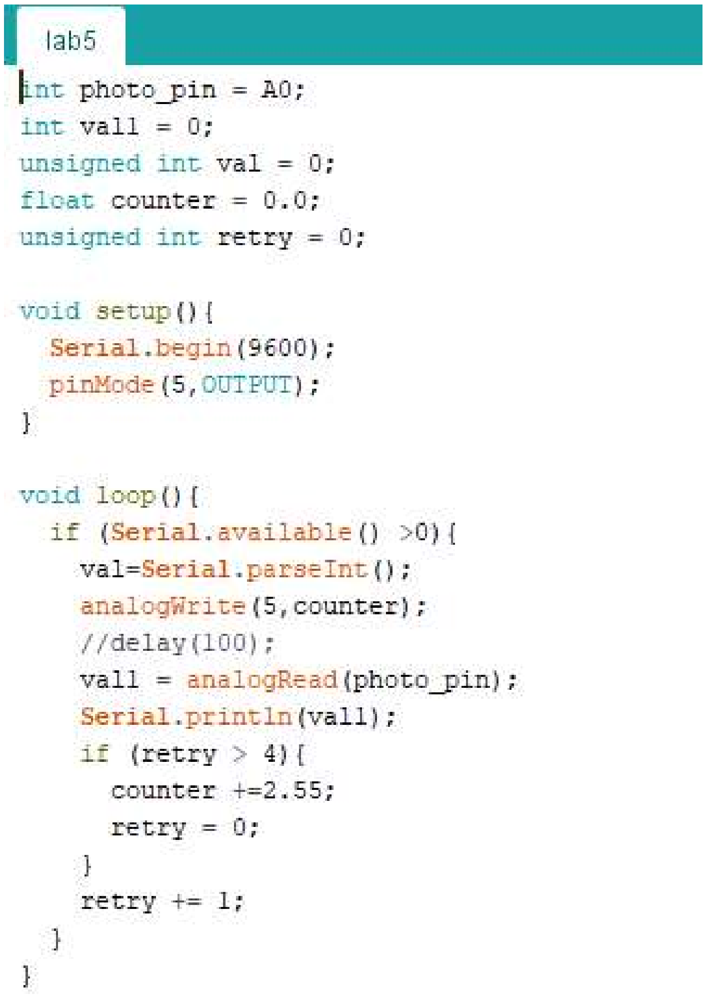


Figure 2. User Interface and Plots Results

Recommendation

From figure 2, we can observe quite a few distortion in the data points which is due to the inconsistency of the testing environment and the performance of the photocell. A slight spot of light can easily alter the data and distort our graph. Those limitation will be responsible for the variety of the data collected. However, we still can observe a consistent result with previous lab 3 and lab 4. The LED current goes down linearly as the duty cycle go up. The photocell voltage is also increasing overtime and getting more saturated when duty cycle is going to 100%. The photocell Resistor also increase with duty cycle. Further experiments with better testing environment and more details data point (1% duty cycle margin, 10 points each iteration for example) needs to be carried out to reduce the distortion and improve the results and analysis.

Arduino Sketch



```
int photo_pin = A0;
int vall = 0;
unsigned int val = 0;
float counter = 0.0;
unsigned int retry = 0;

void setup(){
  Serial.begin(9600);
  pinMode(5,OUTPUT);
}

void loop(){
  if (Serial.available() >0){
    val=Serial.parseInt();
    analogWrite(5,counter);
    //delay(100);
    vall = analogRead(photo_pin);
    Serial.println(vall);
    if (retry > 4){
      counter +=2.55;
      retry = 0;
    }
    retry += 1;
  }
}
```

Python Script

```python
import serial
import matplotlib.pyplot as plt
import time
import numpy
import tkinter as tk
import tkinter.font as tkFont
from tkinter import *
from tkinter import messagebox
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg

plt.close('all')
root = tk.Tk()
root.geometry("1000x600")
root.title("Photoresistor Evaluator")
root.configure(bg='grey')

figure1 = plt.Figure(figsize=(12,6), dpi=100)

ax1 = figure1.add_subplot(231)
fig1 = FigureCanvasTkAgg(figure1,root)
fig1.get_tk_widget().pack(side=tk.LEFT, fill=tk.BOTH)
figure1.set_facecolor('lightgrey')
line1, = ax1.plot([],[],'bo')
ax1.set_xlabel('Duty Cycle (%)', fontsize=10)
#ax1.set_ylabel('Photocell Voltage (V)', fontsize=10)
ax1.set_title('Photocell Voltage (V)')
ax1.grid(b=True, which='major', color='k', linestyle = '-')
ax1.grid(b=True, which='minor', color='r', linestyle = '-', alpha = 0.2)
ax1.minorticks_on()

ax2 = figure1.add_subplot(232)
line2, = ax2.plot([],[],'bo')
ax2.set_xlabel('Duty Cycle (%)',fontsize=10)
#ax2.set_ylabel('Photocell Current (mA)', fontsize=10)
ax2.set_title('Photocell Current (mA)')
ax2.grid(b=True,which='major',color='k',linestyle='-')
ax2.grid(b=True,which='minor',color='r',linestyle='-',alpha=0.2)
ax2.minorticks_on()

ax3 = figure1.add_subplot(233)
line3, = ax3.plot([],[],'bo')
ax3.set_xlabel('Duty Cycle (%)',fontsize=10)
#ax3.set_ylabel('Photocell Resistance (k\u03A9)', fontsize=10)
```

```python
ax3.set_title('Photocell Resistance (k\u03A9)')
ax3.grid(b=True,which='major',color='k',linestyle='-')
ax3.grid(b=True,which='minor',color='r',linestyle='-',alpha=0.2)
ax3.minorticks_on()

ax4 = figure1.add_subplot(235)
line4, = ax4.plot([],[],'bo')
ax4.set_xlabel('Duty Cycle (%)',fontsize=10)
#ax4.set_ylabel('LED Voltage (V)', fontsize=10)
ax4.set_title('LED Voltage (V)')
ax4.grid(b=True,which='major',color='k',linestyle='-')
ax4.grid(b=True,which='minor',color='r',linestyle='-',alpha=0.2)
ax4.minorticks_on()

ax5 = figure1.add_subplot(236)
line5, = ax5.plot([],[],'bo')
ax5.set_xlabel('Duty Cycle (%)',fontsize=10)
#ax5.set_ylabel('LED Current (mA)', fontsize=10)
ax5.set_title('LED Current (mA)')
ax5.grid(b=True,which='major',color='k',linestyle='-')
ax5.grid(b=True,which='minor',color='r',linestyle='-',alpha=0.2)
ax5.minorticks_on()

figure1.subplots_adjust(hspace=0.5)
fontStyle= tkFont.Font(family="Lucida Grande", size=14)
var = StringVar()
label= Label( root,textvariable=var,font=fontStyle,bg='lightgrey')
label.place(x=100,y=400)

def get_valid_avg (buff):
    temp_mean = numpy.mean(buff)
    temp_std = numpy.std(buff)
    temp_buff = []
    for val in buff:
        # Error check, only store valid values to new buffer and return new mean
        if val <= (temp_mean + temp_std) and val >= (temp_mean - temp_std):
            temp_buff.append(val)
    return numpy.mean(temp_buff)

def startCallBack():
    arduino=serial.Serial('COM4',9600,timeout=5)
    time.sleep(1)
    DC=[]
    V_res=[]
    V_pc=[]
    I=[]
    R=[]
    LED_I=[]
    LED_V=[]
    print("***********Lab5***************")
    with open('lab5logs.txt', 'w') as logFile:
        logFile.writelines("***********Lab5***************\n")
        logFile.writelines("Measurements:\n")

        for i in range(100):
            buff_V_pc = []
            DC.append(i)
            var.set("Duty Cycle: {:.2f} %".format(DC[-1]))
            for count in range(5):
                arduino.reset_input_buffer()
                arduino.reset_output_buffer()
                #time.sleep(0.01)
                arduino.write(b'1')
                #time.sleep(0.5)
                #a=int(arduino.read(4).decode("ascii"))
                a = int(arduino.readline().decode("utf-8"))
                v = float(a)*5.0/255.0         # voltage on photocell
                buff_V_pc.append(v)
                print("Iteration: %d, Arduino: %f, Pcell Volt: %f" % (i,a,v))
            # Store output to buffers
            V_pc.append(get_valid_avg(buff_V_pc))
            print ("Duty cycle: %d, Average Voltage Photocell: %f" %(i, get_valid
_avg(buff_V_pc)))
            logFile.writelines("Duty cycle: %d, Average Voltage Photocell: %f \n"
 %(i, get_valid_avg(buff_V_pc)))
            V_res.append(5-V_pc[-1])
            I.append(V_res[-1]*1000.0/1000.0) #mA
            R.append(V_pc[-1]/I[-1])           #kOhm
            vol_R = 5.0-(i*5/100.0)
            LED_I.append(vol_R*1000.0/1000.0)       # mA
            LED_V.append(i*5/100.0)

        # Write logFile
        logFile.writelines("Calculations:\n")
        logFile.writelines("Photocell voltage: 5 * Arduino_reading/255\n")
        logFile.writelines("Photocell Current: (5 - Photocell_vol)/1000 Ohm\n")
        logFile.writelines("Photocell Resistance: Photocell_vol/Photocell_current
\n")
        logFile.writelines("LED Current: (5 - LED_voltage)/1000 Ohm\n")
        logFile.writelines("LED Resistance: LED_voltage/LED_current\n")
        for idx in range(100):
            logFile.writelines("Photocell Voltage: %f, Photocell current: %f, Pho
tocell Resistance: %f, LED Current: %f, LED Resistance: %f\n" % (V_pc[idx], I[idx
], R[idx], LED_I[idx], LED_V[idx]))
        logFile.close()

    line1.set_data(DC,V_pc)
    axa = fig1.figure.axes[0]
    axa.set_xlim(0,100)
    axa.set_ylim(0, 5)

    line2.set_data(DC, I)
    axb = fig1.figure.axes[1]
    axb.set_xlim(0, 100)
    axb.set_ylim(0, 5)

    line3.set_data(DC, R)
    axc = fig1.figure.axes[2]
    axc.set_xlim(0,100)
    axc.set_ylim(0,20)

    line4.set_data(DC, LED_V)
    axd = fig1.figure.axes[3]
    axd.set_xlim(0, 100)
    axd.set_ylim(0, 5)

    line5.set_data(DC, LED_I)
    axe = fig1.figure.axes[4]
    axe.set_xlim(0, 100)
    axe.set_ylim(0, 5)

    fig1.draw()
    plt.pause(0.05)
    plt.show()

    arduino.close()
    return

# Create Start button
start_button = Button(root, text ="Start", font=fontStyle, height=2, width=5, com
mand = startCallBack)
start_button.place(x=100, y=450)
root.mainloop()
```