Image Histograms

Lab #3: Section J

Submitted By: Wen Zhanghao

Submission Date: March 12, 2015

Overview

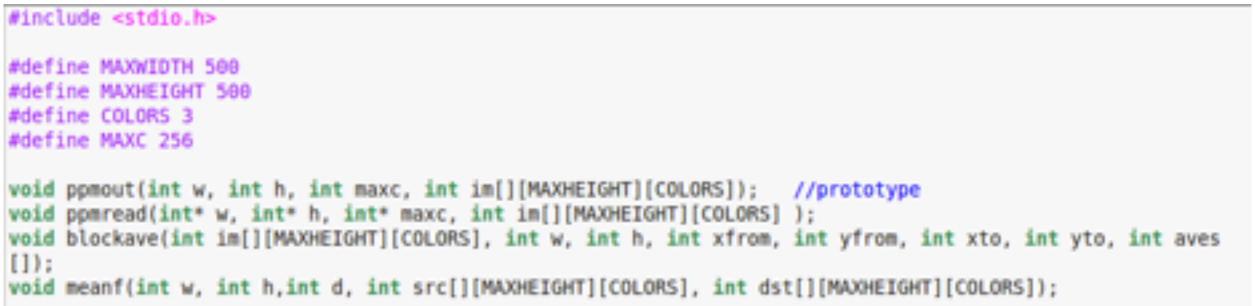
In this lab, we are required to create histograms that represent the frequency of three different color values of given image. And then we apply meanf function to the given original image to create another new image, like the original image goes into a filter.

Basically speaking, in the whole program, we need to complete two functions and other functions are given. one function called imhist that actually count the values and stores this values into corresponding array. The second function needs to be complete called outputhist that is actually print out the data in the arrays that stored frequency to show the histograms.

Analysis

Based on the comment that define which kind of function need us to complete, for the imhist, it gives certain values to each element in one array. And that values is the counts for each color values for every pixel. We have five parameters there, including w — width that defined after ppread function that we don't need to worry about this time. h— represents height that also from the result of ppread, src[][Maxheight][colors]— this is the image we are going to go through this function and color — read different layer of given image (all images 's color is composed of red, green, and blue) and finally, hsit[maxc] — is an array that store those frequency and waits for printing them out when went through function called outputhist. Outputhist function to print the data systematically and should be readable, in the required format.

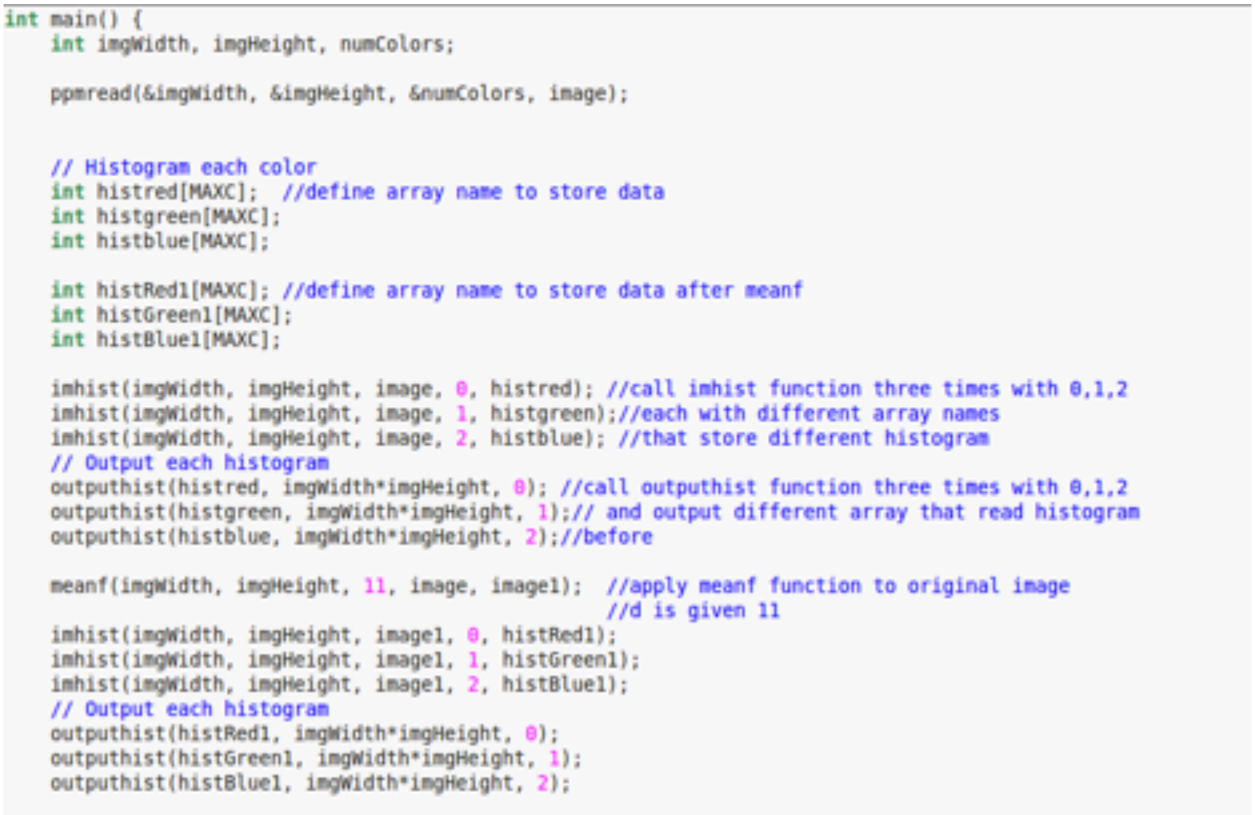
Design



```
#include <stdio.h>

#define MAXWIDTH 500
#define MAXHEIGHT 500
#define COLORS 3
#define MAXC 256

void ppmout(int w, int h, int maxc, int im[][MAXHEIGHT][COLORS]);   //prototype
void ppmread(int* w, int* h, int* maxc, int im[][MAXHEIGHT][COLORS] );
void blockave(int im[][MAXHEIGHT][COLORS], int w, int h, int xfrom, int yfrom, int xto, int yto, int aves
[]);
void meanf(int w, int h,int d, int src[][MAXHEIGHT][COLORS], int dst[][MAXHEIGHT][COLORS]);
```

For this parts, first of all, this program define the maxwidth and maxheight and colors and maxc, so when we initialize the array, we use those name to make sure the accuracy. In addition, the maxwidth and maxheight is 500 that requires the image's size given into this program have to smaller than there two values.

The second part is the prototype, and we will complete the function below the main function. We use prototype to tell the computer that we will use this function later so when computer run the program and meet the name of function, it will not come out an error.

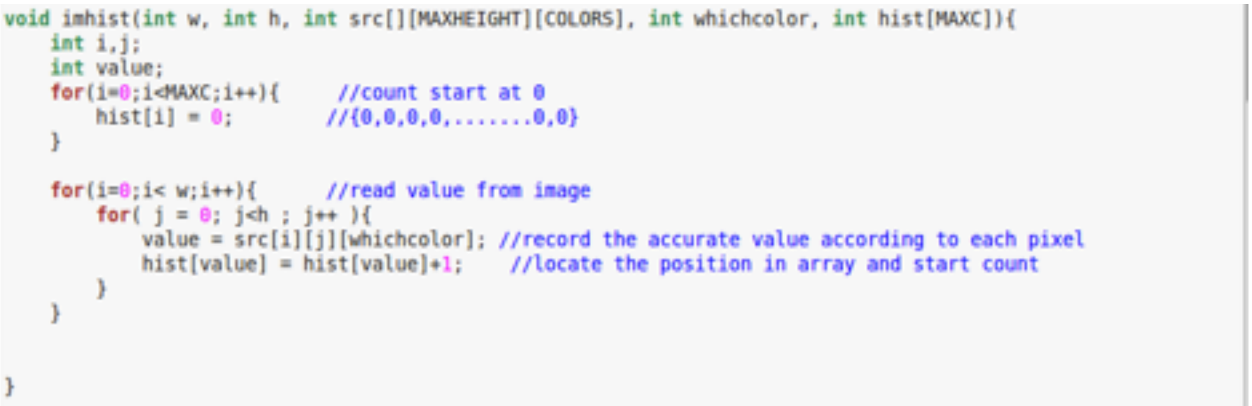


```
int main() {
    int imgWidth, imgHeight, numColors;

    ppmread(&imgWidth, &imgHeight, &numColors, image);


    // Histogram each color
    int histred[MAXC];  //define array name to store data
    int histgreen[MAXC];
    int histblue[MAXC];

    int histRed1[MAXC]; //define array name to store data after meanf
    int histGreen1[MAXC];
    int histBlue1[MAXC];

    imhist(imgWidth, imgHeight, image, 0, histred); //call imhist function three times with 0,1,2
    imhist(imgWidth, imgHeight, image, 1, histgreen);//each with different array names
    imhist(imgWidth, imgHeight, image, 2, histblue); //that store different histogram
    // Output each histogram
    outputhist(histred, imgWidth*imgHeight, 0); //call outputhist function three times with 0,1,2
    outputhist(histgreen, imgWidth*imgHeight, 1);// and output different array that read histogram
    outputhist(histblue, imgWidth*imgHeight, 2);//before

    meanf(imgWidth, imgHeight, 11, image, image1);  //apply meanf function to original image
                                                     //d is given 11
    imhist(imgWidth, imgHeight, image1, 0, histRed1);
    imhist(imgWidth, imgHeight, image1, 1, histGreen1);
    imhist(imgWidth, imgHeight, image1, 2, histBlue1);
    // Output each histogram
    outputhist(histRed1, imgWidth*imgHeight, 0);
    outputhist(histGreen1, imgWidth*imgHeight, 1);
    outputhist(histBlue1, imgWidth*imgHeight, 2);
```

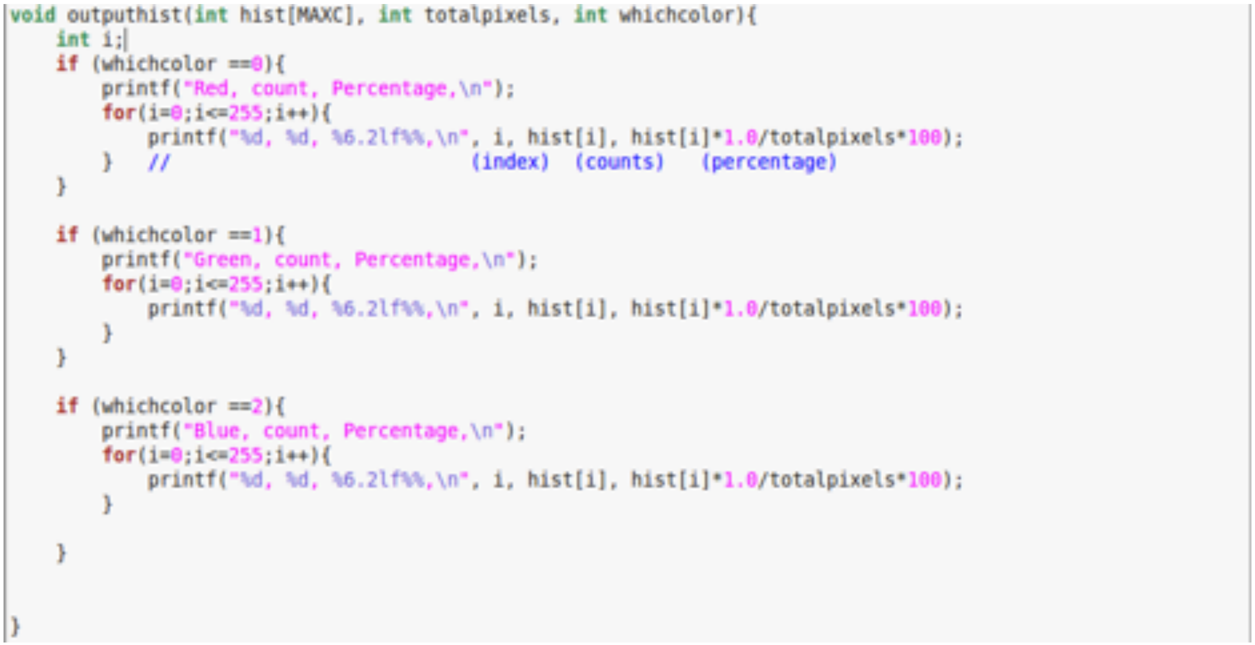
This is the main function part, it calls 13 times function in total, six of it before mean function is to read the frequency and output the frequency of the original image. The other six functions after mean read the frequency and output of the filter image that went through meanf function.

Now let us look one example of how each function works.

ppread sets the variable of imgWidth imgHeight and then Six array's name have been initialized with 256 elements (from 0 to 255). let's put this for example, the imhist (imgWidth, imgHeight, image, 0, histred). In this case, image is src[], and whichcolor is 0 because we read red color values, and histred is the array that store color frequency. I used for loop to set each element in histred is zero and then use nested loop to pick each pixel, and I used variable value to locate



```c
void imhist(int w, int h, int src[][MAXHEIGHT][COLORS], int whichcolor, int hist[MAXC]){
    int i,j;
    int value;
    for(i=0;i<MAXC;i++){      //count start at 0
        hist[i] = 0;         //{0,0,0,0,.......0,0}
    }

    for(i=0;i< w;i++){       //read value from image
        for( j = 0; j<h ; j++ ){
            value = src[i][j][whichcolor]; //record the accurate value according to each pixel
            hist[value] = hist[value]+1;    //locate the position in array and start count
        }
    }


}
```



```c
void outputhist(int hist[MAXC], int totalpixels, int whichcolor){
    int i;
    if (whichcolor ==0){
        printf("Red, count, Percentage,\n");
        for(i=0;i<=255;i++){
            printf("%d, %d, %6.2lf%%,\n", i, hist[i], hist[i]*1.0/totalpixels*100);
        }   //                           (index)  (counts)   (percentage)
    }

    if (whichcolor ==1){
        printf("Green, count, Percentage,\n");
        for(i=0;i<=255;i++){
            printf("%d, %d, %6.2lf%%,\n", i, hist[i], hist[i]*1.0/totalpixels*100);
        }
    }

    if (whichcolor ==2){
        printf("Blue, count, Percentage,\n");
        for(i=0;i<=255;i++){
            printf("%d, %d, %6.2lf%%,\n", i, hist[i], hist[i]*1.0/totalpixels*100);
        }

    }


}
```

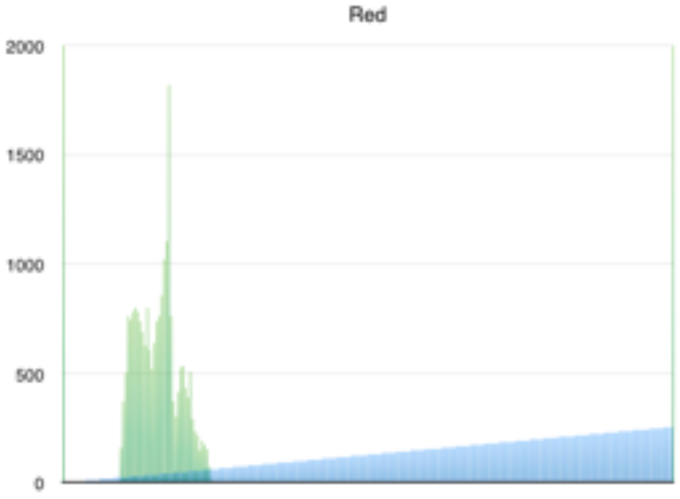
the color value and then start count if it be caught each time. So I changed the histred 's element by counting.

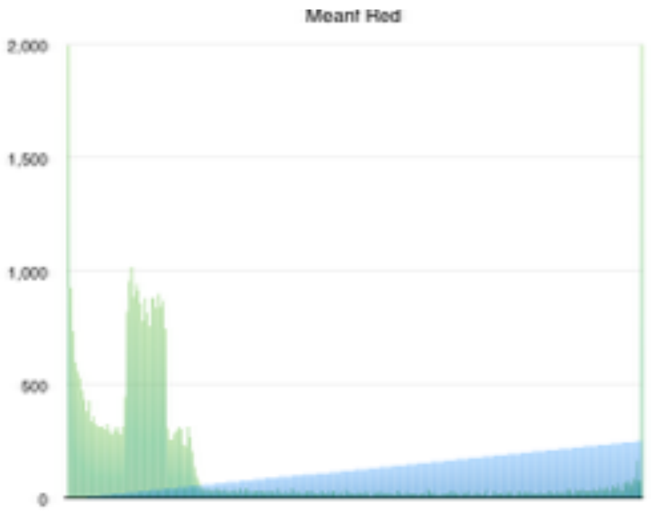
Then we need to output those data, so let look at outputhist function. The first parameter is the array's name that needs to output, total pixel = imgwi * imgheight, and then the third parameter determine whichcolor response to three if statement.

Testing

For the testing parts, I compared my output with the lab manual about halfred output. I redirect input like this: ./ppm < halberd.ppm
and then redirect output like this : ./ppm < halfred.ppm > halfred. csv
One points that need to be mentioned is that the array and total pixel are both int, so when we compute the percentage, we should convert the type into double and correct results.

After redirecting output into .csv file, I open it with spreadsheet and select data to draw the histogram. In the end we get six graphs (attached below), three before meanf and





three after meanf, I contrast graph with the same color, and find that meanf does not have the value that extremely big or extremely small, however, the original graph has some points with 0 frequency or very high frequency. So I guess that the purpose of meanf function is to ‘average’ the values.

Comments

In this lab I leaned a lot about the array especially 3-d array. For reaching each point we have to make like 3 nested-loops.

Another thing I learned a lot is about functions. This is the lab with the most number of functions, and lots of parameters used in each function, which required a lot of logic to analyze and think about.

When we first confront this lab, we are all shocked because there are huge number of things that we don't know. In the future teamwork, we should understand and take advantage of comments to know how the function that created by others works.