浙江大学实验报告

课程名称：图像信息处理 指导老师：宋明黎 成绩：

实验名称：图像对数处理视觉增强与直方图均衡（仅要求灰度图）

一、实验目的和要求

Assignment-3

1. Image logarithmic operation for visibility enhancement
2. Histogram equalization

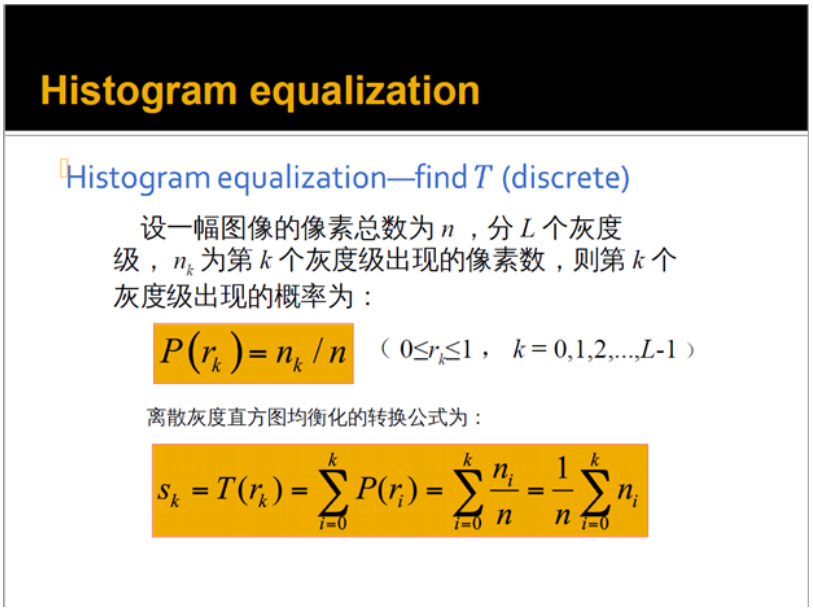
二、实验内容和原理

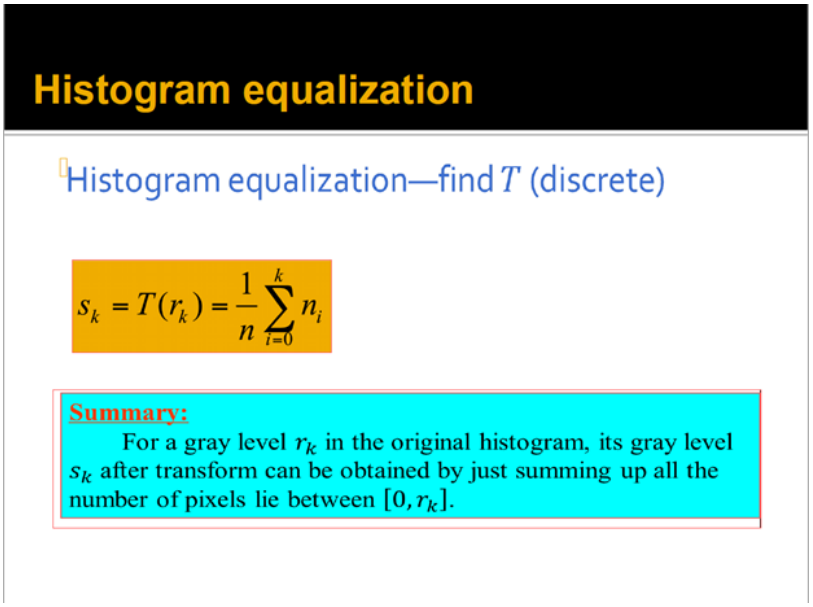
1、图像对数处理

Visibility enhancement: logarithmic operation

- In order to enhance the image's visibility, adjust the pixel value by a logarithmic operator.
- $L_d = \frac{\log(L_w+1)}{\log(L_{max}+1)}$
- L_d is display luminance, L_w is the real luminance, L_{max} is the maximal luminance value in the image.
- This mapping function make sure that: no matter the dynamic range of the scene, the maximal luminance value will be 1 (white), and other values changes smoothly.

2、直方图均衡





三、实验步骤与分析

1、读取 BMP 文件的基本信息

经前两次作业，已可以系统地模块化操作了，直接使用函数 ReadBmp。由于只要求灰度图的处理，因此将彩色图一并顺带转化为灰度图，方便操作。其中需要保存入全局变量的有：Clr 颜色数，bitcount 文件的位，PW 数据的宽度，PH 数据的高度，QUAD 颜色版，pSize 图像数据大小，LineSize 行数据大小，数组 P 保存图像数据。

2、对数处理

上次实验灰度化才用 3 个相同字节一存的 24 位图，本次实验打算实现正规的灰度化，将灰度 bmp 存为 8 位图。

```
void Logarithmic(char* b)
{
    int max=0,index;
    if (Clr == 0)
        for (int i = 0; i < pH; i++)//行
            if (bitCount == 24)
            {
                for (int j = 0; j < pW*3; j = j + 3)// 列
                {
                    int n = i*LineSize + j;
                    index = P[n] = P[n + 1] = P[n + 2] = (BYTE)(0.299 * P[n] + 0.587 * P[n + 1] + 0.114 * P[n + 2]); //分别是r,g,b分量
                    if(index > max)
                        max = index;
                }
            }
//    printf("%d\n",max);
    for (int i = 0; i < pH; i++)//行
        if (bitCount == 24)
            for (int j = 0; j < pW * 3; j = j + 3)// 列
            {
                int n = i * LineSize + j;
                P[n] = P[n + 1] = P[n + 2] = (BYTE)(255 * log10( _X P[n] + 1) / log10( _X max + 1));
            }
    FILE* dFile = fopen(b, _Mode: "wb");//创建目标文件
    fwrite(&fileHeader, sizeof(BITMAPFILEHEADER), _Count: 1, dFile);   //写入文件头
    fwrite(&infoHeader, sizeof(BITMAPINFOHEADER), _Count: 1, dFile);  //写入文件信息头
    if (QUAD)
        fwrite(QUAD, _Size: sizeof(RGBQUAD)* Clr, _Count: 1, dFile);
    //写入图像数据
    fwrite(P, pSize, _Count: 1, dFile);
    fclose(dFile);
    if (QUAD)
    {
        free(QUAD);
        QUAD = NULL;
    }
    if (P)
    {
        free(P);
        P = NULL;
    }
}
```

首先将颜色值 rgb 灰度化后填入数据数组 P，随后对此利用公式进行对数操作，然后保存如目标文件即可

2、直方图

```
void Histogram(char* b)
{
    int Min = 120;
    int Max = 120;
    int now;
    double grey[256]={0};
    if (Clr == 0)
        for (int i = 0; i < pH; i++)//行
            if (bitCount == 24)
                for (int j = 0; j < pW * 3; j = j + 3)// 列
                {
                    int n = i * LineSize + j;
                    now = P[n] = P[n + 1] = P[n + 2] = (BYTE)(0.299 * P[n] + 0.587 * P[n + 1] + 0.114 * P[n + 2]); //分别是r,g,b分量
                    grey[P[n]]++;       //灰度直方图
                    if(now > Max)
                        Max = now;           //找到最大最小
                    else if(now < Min)
                        Min = now;
                }
    for(int i = 0; i < Max; i++)
        grey[i] = grey[i] / (pH * pW);
//    printf("max=%d,min=%d\n", Max, Min);
    //累计直方图
    double greyadd[256]={0};
    for(int i = 0; i < 253; i++)
        for(int j = 0; j <= i; j++)
            greyadd[i] += grey[j];
//    printf("%f %f", greyadd[0], greyadd[1]);
//    printf("\n%f\n%f\n%f", grey[0], grey[1], grey[253]);

    //新值
    if (Clr == 0)
        for (int i = 0; i < pH; i++)//行
            if (bitCount == 24)
                for (int j = 0; j < pW * 3; j = j + 3)// 列
                {
                    int n = i * LineSize + j;
                    P[n] = P[n + 1] = P[n + 2] = (BYTE)(Min + (Max - Min) * greyadd[P[n]]);
                }
```

```
    FILE* dFile = fopen(b,  _Mode: "wb");//创建目标文件
    fwrite(&fileHeader, sizeof(BITMAPFILEHEADER),  _Count: 1, dFile);   //写入文件头
    fwrite(&infoHeader, sizeof(BITMAPINFOHEADER),  _Count: 1, dFile);  //写入文件信息头
    if (QUAD)
        fwrite(QUAD,  _Size: sizeof(RGBQUAD) * Clr,  _Count: 1, dFile);
    //写入图像数据
    fwrite(P, pSize,  _Count: 1, dFile);
    fclose(dFile);
    if (QUAD)
    {
        free(QUAD);
        QUAD = NULL;
    }
    if (P)
    {
        free(P);
        P = NULL;
    }
}
```

同上，存入后利用直方图算法计算，保存如目标文件

3、调用与结束

```
int main()
{
    char *filename;
    filename = (char*) malloc( _Size: 100 * sizeof(char));
    printf( _Format: "请输入文件路径：");
    scanf( _Format: "%s", filename);
    ReadBmp(filename);
    Logarithmic( b: "lgc.bmp");
    ReadBmp(filename);
    Histogram( b: "Hc1.bmp");
    free(filename);
    return 0;
}
```

注意关闭文件

四、实验环境及运行方法

编译环境：C 语言，使用最新 C11 标准。ide 使用 clion，利用 cmakelist 编译。用 dev 可直接打开 main.c 源文件进行编译。由于 clion 本身默认会对中文输出乱码，故修改为 UTF-8 解码及 GBK 编码，dev 打开可能出现乱码，visual studio 打开无影响

测试方法：断点单步测试、与 matlab 结果相比较

五、实验结果展示

原图

对数处理

直方图均衡

生成的图片与编译文件在同一路径下，使用 clion 时在 cmake-build-debug 目录下

六、心得体会

本次作业利用的数学公式更为复杂，对编码有点挑战。