Homework2 图片旋转

姓名：胡成成

学号：[redacted]

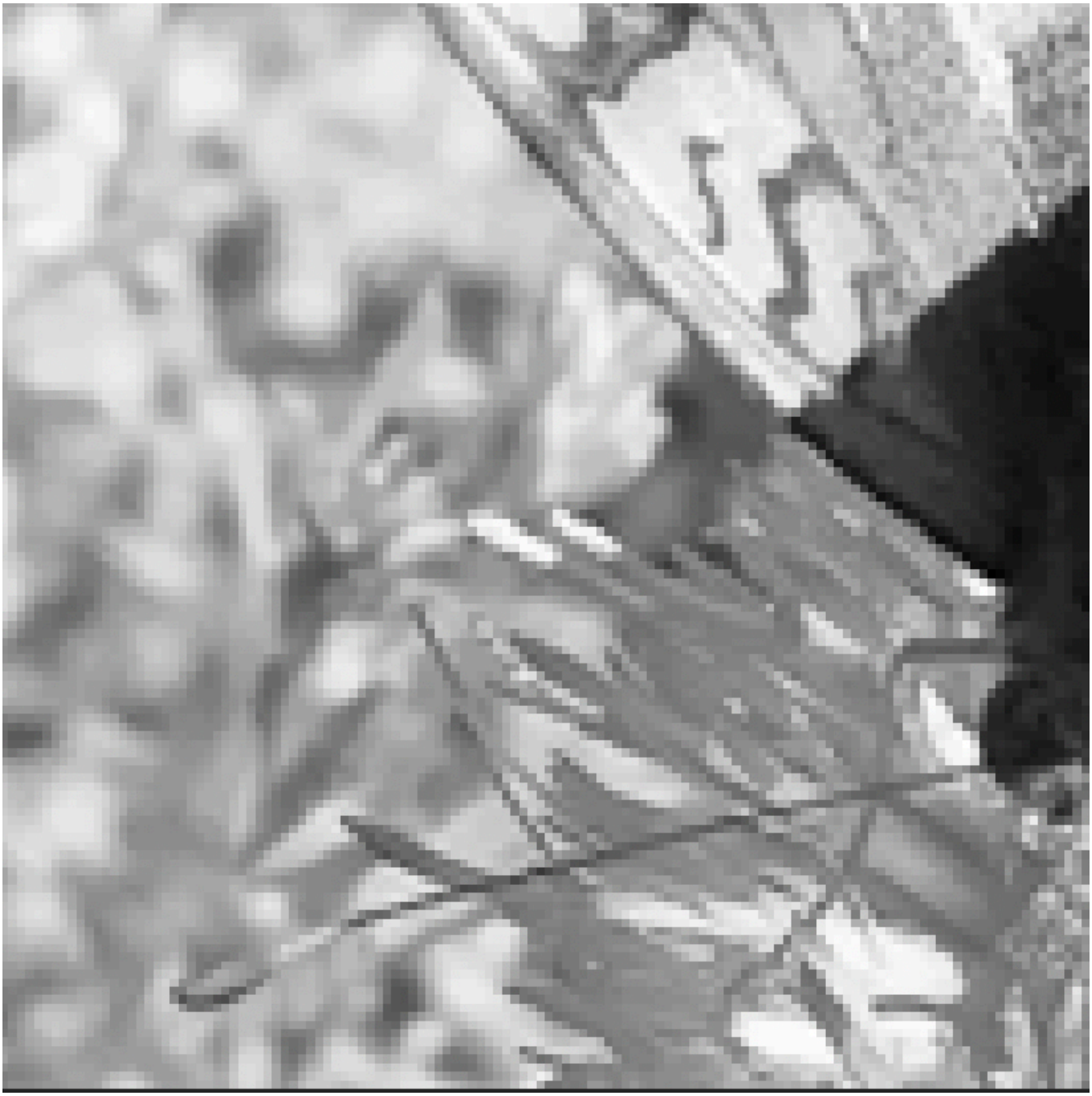
Problem

导入一幅256X256的8bit灰度图像，请在ARM 处理器上编程，使图像顺时针旋转45度，并导出图像

Answer

过程分析

1. 首先读取图片观察，如下图所示：



2. 其次，我们需要知道raw格式文件的读入读出方式：`fread` 和`fwrite`函数
3. 能够读入读出后，我们需要知道矩阵旋转的数学原理：
    - 首先需要了解旋转矩阵：

$$\begin{bmatrix} \alpha & \beta & (1-\alpha)\cdot \mathtt{center.x} - \beta \cdot \mathtt{center.y} \\ -\beta & \alpha & \beta \cdot \mathtt{center.x} + (1-\alpha)\cdot \mathtt{center.y} \end{bmatrix}$$

$$\begin{aligned} \alpha &= \mathtt{scale} \cdot \cos \mathtt{angle}, \\ \beta &= \mathtt{scale} \cdot \sin \mathtt{angle} \end{aligned}$$

    - 其次是坐标变换关系：

$$\mathtt{dst}(x,y) = \mathtt{src}(M_{11}x + M_{12}y + M_{13}, M_{21}x + M_{22}y + M_{23})$$

详细代码及解释

1. 导入必要的库

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
```

2. 定义初始化参数：图片大小，常数和图片路径并读入

```
#define PI 3.1415926

#define height  128
#define width 128

typedef unsigned char  BYTE;  // Define Byte type , using space 1 byte

FILE *fp = NULL;         // Using to read file

  BYTE Input[height][width];  // Store image pixel

  // Raw image path: input and output
  char path[256] = "/Users/admin/Downloads/butter128.raw";
  char outpath[256] = "/Users/admin/Downloads/butter128out.raw";

  int i,j;

  // Read image
  printf("Start reading raw image!\n");
  if((fp = fopen( path, "rb" )) == NULL)
  {
      printf("Can not open the raw image!\n" );
      return 0;
  }
  else
        {
            printf("Read image OK!\n");
        }

  for( i = 0; i < height; i++ )
  {
            for( j = 0; j < width ; j ++ )
    {
                fread( &Input[i][j], 1, 1, fp );
        //printf("%d  \t",Input[i][j]);       //print all piexl
    }
  }
```

```
        printf("Read all pixel!\n");
  fclose(fp);
```

3. 图片旋转与公式参数定义

```
// rotate image
        double angle = -45;   //giving a anger +: 逆时针

        double sita = angle*PI / 180;

        // get the new size of output image
        double a = (width - 1) / 2.0;
  double b = (height - 1) / 2.0;

  double x1 = -a*cos(sita) - b*sin(sita);
  double y1 = -a*sin(sita) + b*cos(sita);

  double x2 = a*cos(sita) - b*sin(sita);
  double y2 = a*sin(sita) + b*cos(sita);

  double x3 = a*cos(sita) + b*sin(sita);
  double y3 = a*sin(sita) - b*cos(sita);

  double x4 = -a*cos(sita) + b*sin(sita);
  double y4 = -a*sin(sita) - b*cos(sita);

  int wo = round(fmax(abs(x1 - x3), abs(x2 - x4)));
  int ho = round(fmax(abs(y1 - y3), abs(y2 - y4)));

        double centerX = (width+1)/2.0;
        double centerY = (height+1)/2.0;

        // init convert parameters
        double alph = cos(sita);
        double beta = sin(sita);

        double M11 = alph;
        double M12 = beta;
        double M13 = (1-alph)*centerX-beta*centerY;
        double M21 = -beta;
        double M22 = alph;
        double M23 = (1-alph)*centerY+beta*centerX;

        BYTE Output[height][width];
        //BYTE Output[ho][wo];
```

4. 旋转图片并导出

```
// Write image
        printf("Start rotating and writing raw image!\n");
  if( ( fp = fopen( outpath, "wb" ) ) == NULL )
  {
      printf("Can not create the raw_image : %s\n", outpath );
      return 0;
  }

        for( i = 0; i < height; i++ )
  {
      for( j = 0; j < width; j ++ )
    {
                // rotate image
                int ox = round(M11*i+M12*j+M13);
                int oy = round(M21*i+M22*j+M23);

                // solve the egde
                if(ox < 0 || ox > width-1 || oy < 0 || oy > height-1){
                    Output[i][j] = 0;
                }
                else{
                    Output[i][j] = Input[ox][oy];
                }
                // write to raw image
    fwrite( &Output[i][j], 1 , 1, fp );
    }
  }

        printf("Write image OK!\n");

  fclose(fp);
```

全部代码

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>

#define PI 3.1415926

#define height  128
#define width 128

typedef unsigned char  BYTE;  // Define Byte type , using space 1 byte

int main()
{
```

```
FILE *fp = NULL;        // Using to read file

BYTE Input[height][width];  // Store image pixel

// Raw image path: input and output
char path[256] = "/Users/admin/Downloads/butter128.raw";
char outpath[256] = "/Users/admin/Downloads/butter128out.raw";

int i,j;

// Read image
printf("Start reading raw image!\n");
if((fp = fopen( path, "rb" )) == NULL)
{
    printf("Can not open the raw image!\n" );
    return 0;
}
else
      {
          printf("Read image OK!\n");
      }

for( i = 0; i < height; i++ )
{
          for( j = 0; j < width ; j ++ )
  {
              fread( &Input[i][j], 1, 1, fp );
      //printf("%d  \t",Input[i][j]);       //print all piexl
  }
}
      printf("Read all pixel!\n");
fclose(fp);

// rotate image
      double angle = -45;   //giving a anger +: 逆时针

      double sita = angle*PI / 180;

      // get the new size of output image
      double a = (width - 1) / 2.0;
double b = (height - 1) / 2.0;

double x1 = -a*cos(sita) - b*sin(sita);
double y1 = -a*sin(sita) + b*cos(sita);

double x2 = a*cos(sita) - b*sin(sita);
double y2 = a*sin(sita) + b*cos(sita);

double x3 = a*cos(sita) + b*sin(sita);
```

```
double y3 = a*sin(sita) - b*cos(sita);

double x4 = -a*cos(sita) + b*sin(sita);
double y4 = -a*sin(sita) - b*cos(sita);

int wo = round(fmax(abs(x1 - x3), abs(x2 - x4)));
int ho = round(fmax(abs(y1 - y3), abs(y2 - y4)));

      double centerX = (width+1)/2.0;
      double centerY = (height+1)/2.0;

      // init convert parameters
      double alph = cos(sita);
      double beta = sin(sita);

      double M11 = alph;
      double M12 = beta;
      double M13 = (1-alph)*centerX-beta*centerY;
      double M21 = -beta;
      double M22 = alph;
      double M23 = (1-alph)*centerY+beta*centerX;

      BYTE Output[height][width];
      //BYTE Output[ho][wo];

      // Write image
      printf("Start rotating and writing raw image!\n");
if( ( fp = fopen( outpath, "wb" ) ) == NULL )
{
    printf("Can not create the raw_image : %s\n", outpath );
    return 0;
}

      for( i = 0; i < height; i++ )
{
    for( j = 0; j < width; j ++ )
  {
              // rotate image
              int ox = round(M11*i+M12*j+M13);
              int oy = round(M21*i+M22*j+M23);

              // solve the egde
              if(ox < 0 || ox > width-1 || oy < 0 || oy > height-1){
                  Output[i][j] = 0;
              }
              else{
                  Output[i][j] = Input[ox][oy];
              }
              // write to raw image
```

```
    fwrite( &Output[i][j], 1 , 1, fp );
    }
  }

        printf("Write image OK!\n");

  fclose(fp);

        return 0;
}
```

程序结果

输入给定的图片，我们便可以得到旋转任意角度的图片，这里以45度顺时针结果为例：