深 圳 大 学 实 验 报 告

课程名称：数字图像处理

实验项目名称：图像的几何变换

学院：计算机与软件学院

专业：计算机科学与技术

指导教师：吴惠思 教授

报告人：林浩晟 学号：2022280310

实验时间：2025 年 6 月 2 日

实验报告提交时间：2025 年 6 月 2 日

教务部制

一、实验目的：

1) 掌握图像简单几何变换原理
2) 掌握图像简单几何变换实现方法
3) 掌握图像平移变换实现方法
4) 掌握图像缩放变换实现方法
5) 掌握图像旋转变换实现方法

二、实验原理：

实验要求

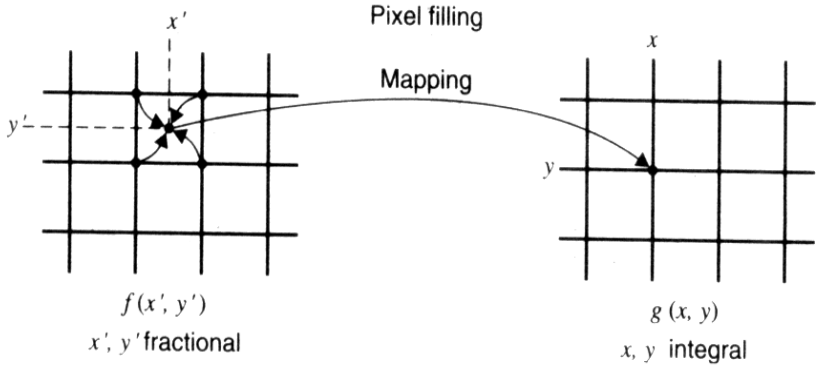
1) 熟悉 C++语言编程
2) 熟练使用 C++语言实现图像文件的读取操作
3) 熟练使用 C++语言实现图像显示方法

实验内容

1、实现图像简单几何变换函数
2、实现图像平移变换函数
3、实现图像缩放变换函数
4、实现图像旋转变换函数

1、图像简单几何变换函数

向后映射法：将输出图像中的每个像素逐一映射回到输入图像中的相应位置
由围住该位置的四个输入图像像素中的左上角像素值作为该输出图像像素的灰度值(简单起见)



空间变换的一般方程为：

$$g(x,y) = f(x',y') = f[a(x,y), b(x,y)]$$

将 a(x,y)、b(x,y)与 x, y 关系写成矩阵形式,即

$$\begin{pmatrix} a(x,y) \\ b(x,y) \\ 1 \end{pmatrix} = \begin{pmatrix} r_{(0)} & s_{(1)} & t_{(2)} \\ u_{(3)} & v_{(4)} & w_{(5)} \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

r, s, t; u, v, w 取不同的值，将得到不同的空间变换

2、图像平移变换函数

- a(x,y) = x + x0
- b(x,y) = y + y0

$$\begin{pmatrix} a(x,y) \\ b(x,y) \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

3、图像缩放变换函数

- a(x,y) = (1/c) * x
- b(x,y) = (1/d) * y

$$\begin{pmatrix} a(x,y) \\ b(x,y) \\ 1 \end{pmatrix} = \begin{pmatrix} 1/c & 0 & 0 \\ 0 & 1/d & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

4、图像旋转变换函数

- a(x,y) = x cos(θ) - y sin(θ)
- b(x,y) = x sin(θ) + y cos(θ)

$$\begin{pmatrix} a(x,y) \\ b(x,y) \\ 1 \end{pmatrix} = \begin{pmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

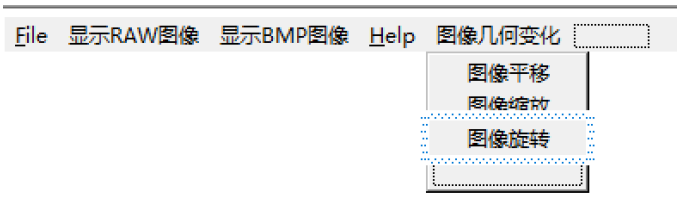
三、实验用品：

计算机、visual C++6.0

五、实验现象及数据处理：

1. 首先在 Menu 处添加选项；设置 ID 为 IDM_1，标明为“图像平移”。

2. 如图添加所有的选项；包括“图像平移”、“图像缩放”、“图像旋转”。



3. 添加图像简单几何变换函数；其中 sgt_Array 为 3×3 仿射变换矩阵，以一维数组形式传入，共 9 个元素；① 首先清空输出图像，将目标图像全部初始化为黑色（像素值 0），为后续写入做准备；② 随后遍历目标图像的每个像素点 (sx, sy)；③ 再将目标图像坐标从左上角原点 (0,0) 转换为图像中心为原点的坐标系，方便应用仿射矩阵（因为仿射通常以图像中心为变换基点）；④ 接下来应用仿射变换矩阵，同时将变换后的位置从中心原点坐标还原回图像左上角为原点的坐标系；⑤ 最后判断是否越界，如果该位置有效，则采样原图像素；否则设为黑色。

```
void SimpleGeometricTranslation(char *oImage, char *nImage, int wImage, int hImage,
                        double *sgt_Array)
{   double axy, bxy;        int i, sx, sy, ox, oy;
    for (i=0; i<wImage*hImage; i++) nImage[i] = 0;        //清空新图像
    for (sy=0; sy<hImage; sy++) {
        for (sx=0; sx<wImage; sx++) {
          ox  = sx - wImage/2;                  //以图像中心为原点
          oy  = sy - hImage/2;
          axy = sgt_Array[0]*(double)ox + sgt_Array[1]*(double)oy + sgt_Array[2];
          bxy = sgt_Array[3]*(double)ox + sgt_Array[4]*(double)oy + sgt_Array[5];
          axy += wImage/2;                  //恢复图像原点位置
          bxy += hImage/2;
          if ((axy<0.0) || (axy>=wImage-1) || (bxy<0.0) || (bxy>=hImage-1))
               nImage[sy*wImage+sx] = 0;              //超出范围部分置0
          else
               nImage[sy*wImage+sx] = oImage[((int)bxy)*wImage+(int)axy];
}}}
```

4. 添加图像平移函数；其中 xPos 代表水平方向的平移距离（+向右，−向左），yPos 代表垂直方向的平移距离（+向下，−向上）；① 首先初始化仿射变换矩阵；② 接下来将仿射变换矩阵中的平移项设置为 xPos 和 yPos，变换矩阵变为：

$$\begin{bmatrix} 1 & 0 & xPos \\ 0 & 1 & yPos \\ 0 & 0 & 1 \end{bmatrix}$$

这意味着图像中每个像素点（x，y）将被变换为（x + xPos，y + yPos），即完成平移；③ 最后调用图像变换函数完成平移处理。

```
void ImageTranslation(char *oImage, char *nImage, int wImage, int hImage,
                      int xPos, int yPos)
{   double gt_Array[9] =    {1, 0, 0,
                    0, 1, 0,
                    0, 0, 1};

    gt_Array[2] = xPos;              //水平方向平移距离
    gt_Array[5] = yPos;              //垂直方向平移距离

    SimpleGeometricTranslation(oImage, nImage, wImage, hImage, gt_Array);
}
```

5. 添加图像缩放函数；xScaling 代表水平方向的缩放倍数（如 2.0 表示放大 2 倍，0.5 表示缩小一半），yScaling 代表垂直方向的缩放倍数；① 首先构建变换矩阵，如下：

$$\begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ 0 & 0 & 1 \end{bmatrix}$$

② 随后设置 a00 = 1 / xScaling，a11 = 1 / yScaling；因为 SimpleGeometricTranslation 函数是反向映射，从输出图像的位置（sx，sy）推回原图中应采样的坐标（axy，bxy），所以缩放时用的是 1 / 缩放倍数；其中 a00 表示水平方向缩放，a01 表示垂直方向缩放；而其余项都为 0，表示不涉及其他变化；③ 最后调用图像变换函数完成缩放处理。

```
void ImageScaling(char *oImage, char *nImage, int wImage, int hImage,
                  double xScaling, double yScaling)
{   double gt_Array[9] =    {0, 0, 0,
                    0, 0, 0,
                    0, 0, 1};

    gt_Array[0] = 1.0 / xScaling;        //水平方向放大倍数
    gt_Array[4] = 1.0 / yScaling;        //垂直方向放大倍数

    SimpleGeometricTranslation(oImage, nImage, wImage, hImage, gt_Array);
}
```

6. 添加图像旋转函数；其中 iAngle 表示旋转角度（单位为“度”，例如 90 表示逆时针旋转 90°）；① 首先初始化仿射变换矩阵，形式为：

$$\begin{bmatrix} a & b & tx \\ c & d & ty \\ 0 & 0 & 1 \end{bmatrix}$$

② 再将角度值转为弧度，因为 cos 和 sin 函数的参数单位是弧度；③ 随后构建旋转矩阵，其中 a=cos(iAngle)，b=-sin(iAngle)，c=sin(iAngle)，d=cos(iAngle)；同时由于 SimpleGeometricTranslation 的坐标计算中已经将图像中心作为坐标原点（ox = sx - wImage/2），所以这里无需手动设置平移项 tx、ty，函数内部已经处理了中心旋转；④ 最后调用图像变换函数完成旋转处理。

```
void ImageRotation(char *oImage, char *nImage, int wImage, int hImage,
                   double iAngle)
{   double pi = 3.14159;
    double gt_Array[9] =    {0, 0, 0,
                0, 0, 0,
                0, 0, 1};
    iAngle = (iAngle/360)*2*pi;
    gt_Array[0] = cos(iAngle);
    gt_Array[1] =-sin(iAngle);
    gt_Array[3] = sin(iAngle);
    gt_Array[4] = cos(iAngle);

    SimpleGeometricTranslation(oImage, nImage, wImage, hImage, gt_Array);
}
```

7. 在程序开头导入<cmath>库，用于使用 sin、cos 函数；同时在开头声明新添加的函数；最后定义 NewImage，大小与 OrgImage 一致，用于储存处理后的图像。

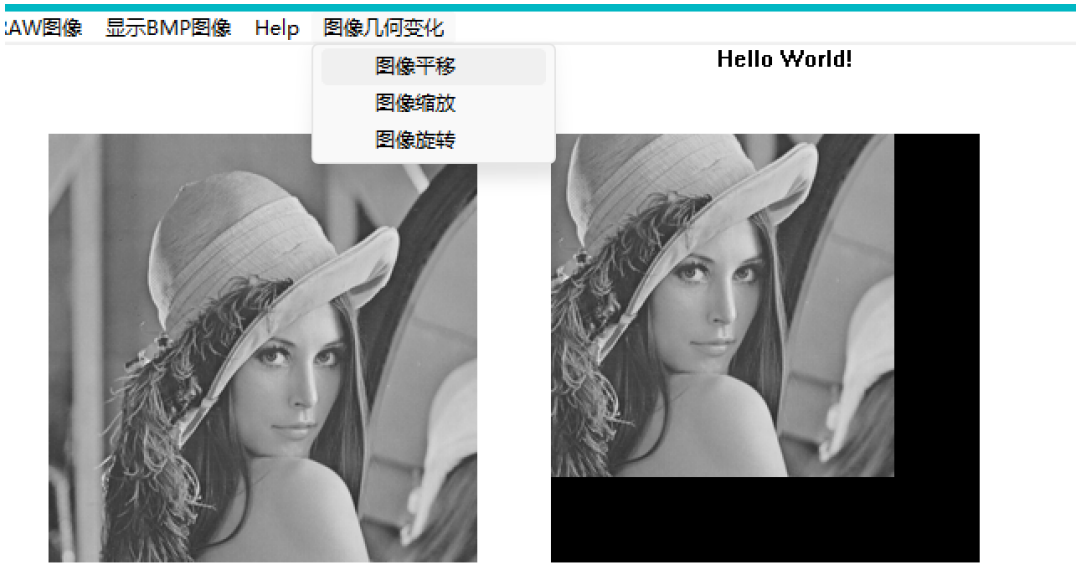
```
#include "stdafx.h"
#include "resource.h"
#include <direct.h>
#include <commdlg.h>
#include <cmath>
#define MAX_LOADSTRING 100

BOOL ReadImage(LPSTR, char *, int, int); //读取图像信息并保存在Image[][]中
void ShowImage(char *, int, int, int, int);
BOOL ReadBmpImage(LPSTR, char *);
void ShowBmpImage(char *, int, int, int, int);
void OpenImageFileDlg(char *);
void SimpleGeometricTranslation(char *oImage, char *nImage, int wImage, int hImage,
                     double *sgt_Array);
void ImageTranslation(char *oImage, char *nImage, int wImage, int hImage,
                   int xPos, int yPos);
void ImageScaling(char *oImage, char *nImage, int wImage, int hImage,
              double xScaling, double yScaling);
void ImageRotation(char *oImage, char *nImage, int wImage, int hImage,
                   double iAngle);
HDC  hWinDC;
int ImageWidth, ImageHeight;
char ImgDlgFileName[MAX_PATH];
char ImgDlgFileDir[MAX_PATH];
char OrgImage[1024*1024];
char NewImage[1024*1024];
#define IMAGEWIDTH  256
#define IMAGEHEIGHT 256
```

8. 在主函数中加入变换选项，其中 case IDM_1 表示的是图像平移，内部先对 OrgImage 进行平移处理，再将 NewImage 即处理后的图像进行显示；case IDM_2 表示图像缩放，case IDM_3 表示图像旋转。

```
        wmEvent = HIWORD(wParam);
        // Parse the menu selections:
        switch (wmId)
        {
            case IDM_SHOWRAWIMAGE:
                OpenImageFileDlg("打开图像文件");
                ReadImage(ImgDlgFileName, OrgImage, IMAGEWIDTH, IMAGEHEIGHT);
                ShowImage(OrgImage, IMAGEWIDTH, IMAGEHEIGHT, XPOS, YPOS);
               break;
            case IDM_SHOWBMPIMAGE:
                OpenImageFileDlg("打开图像文件");
                ReadBmpImage(ImgDlgFileName, OrgImage);
                ShowBmpImage(OrgImage, ImageWidth, ImageHeight, XPOS, YPOS);
               break;
            case IDM_1:
                ImageTranslation(OrgImage, NewImage, IMAGEWIDTH, IMAGEHEIGHT,
              50, 50);
                ShowImage(NewImage, IMAGEWIDTH, IMAGEHEIGHT, XPOS, YPOS+300);
                break;
            case IDM_2:
                ImageScaling(OrgImage, NewImage, IMAGEWIDTH, IMAGEHEIGHT,
          (double)1.5,(double)1.5);
                ShowImage(NewImage, IMAGEWIDTH, IMAGEHEIGHT, XPOS, YPOS+300);
                break;
            case IDM_3:
                ImageRotation(OrgImage, NewImage, IMAGEWIDTH, IMAGEHEIGHT,
               30);
                ShowImage(NewImage, IMAGEWIDTH, IMAGEHEIGHT, XPOS, YPOS+300);
                break;
            case IDM_ABOUT:
               DialogBox(hInst, (LPCTSTR)IDD_ABOUTBOX, hWnd, (DLGPROC)About);
               break;
            case IDM_EXIT:
               DestroyWindow(hWnd);
               break:
```

9. 运行结果如下；

图像平移



图像缩放


RAW图像
显示BMP图像
Help
图像几何变化
图像平移
图像缩放
图像旋转
Hello World!

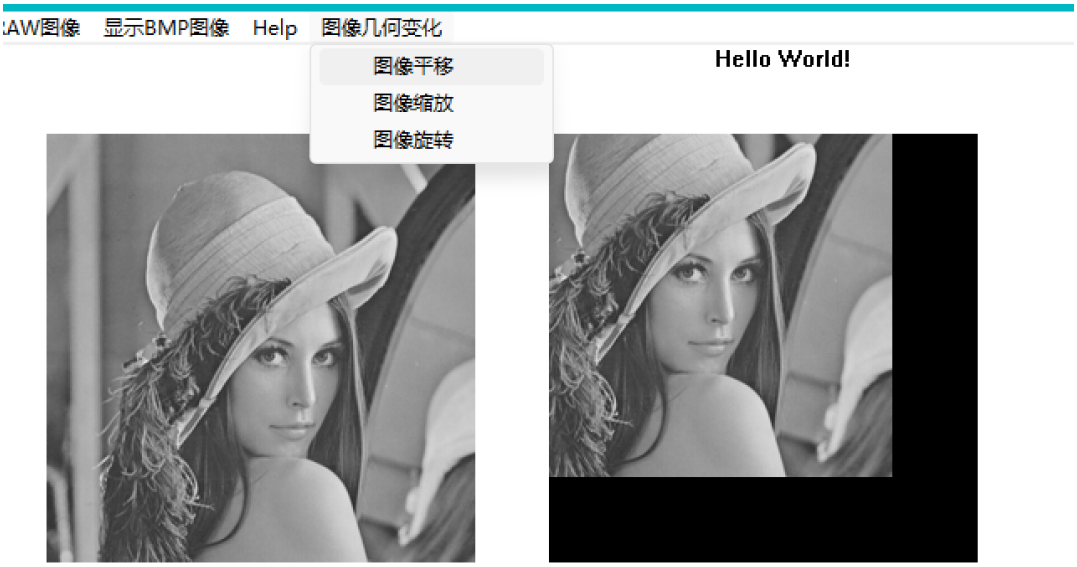
图像旋转


AW图像
显示BMP图像
Help
图像几何变化
图像平移
图像缩放
图像旋转
Hello World!

六、实验结论：

实验成功完成！ 成功实现了图像简单几何变换函数、图像平移变换函数、图像缩放变换函数、实现图像旋转变换函数，运行结果如下；

图像平移



图像缩放

图像旋转

实验感想：

通过本次实验，我对 C++语言在图像处理领域的应用有了更深入的理解。在实现图像简单几何变换、平移、缩放和旋转的过程中，我熟练掌握了 C++语言的相关编程技巧，尤其是文件读取和图像显示方法的实现。实验中遇到的困难，如坐标转换和矩阵运算的准确性，促使我不断查阅资料、调试代码，最终成功解决了问题。这不仅提升了我的编程能力，也让我对图像处理的原理有了更直观的认识。同时，我也意识到细节处理的重要性，一个小错误可能导致整个程序无法正常运行。这次实验让我更加自信地面对复杂编程任务，也激发了我对图像处理技术进一步探索的兴趣。

思考题： 无
指导教师批阅意见： **成绩评定：** **指导教师签字：** **年　　月　　日**
备注：

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。