第一次实验报告

09014217 刘兴成

1.读片软件的功能设计

该读片软件具有读取医学图像裸数据并显示在屏幕上的功能，同时使用者可以通过鼠标拖拽图片位置查看图片各个部位，并且可以通过鼠标滚轮滑动控制图片放大缩小。

2.选择算法依据

所需要用到算法的地方主要为图像的缩小和放大以及图像的移动三部分，其中缩小图像部分采用的算法是隔行提取，而放大图像部分采用的插值算法是最邻近元法。其中对于放大图像来说，因为医学图像尺寸较大，所以采用运算速度最快的最邻近元法来做处理，与其他插值算法相比能够获得更快的处理速度。

2.1 缩小图像算法

```
testimg.Destroy();
testimg=CImage();
testimg.Create(int(width/div),int(height/div),24);//创建缩小之后的图像尺寸
for(int i=0;i<int(width/div);i++)
    for(int j=0;j<int(height/div);j++)
        {
            unsigned int num=buffer[j*(int)div*(int)width+int(float(i)*div)];//隔行提取像素灰度值
            testimg.SetPixel(i,j,RGB(num,num,num));
        }
```

2.2 最邻近元算法

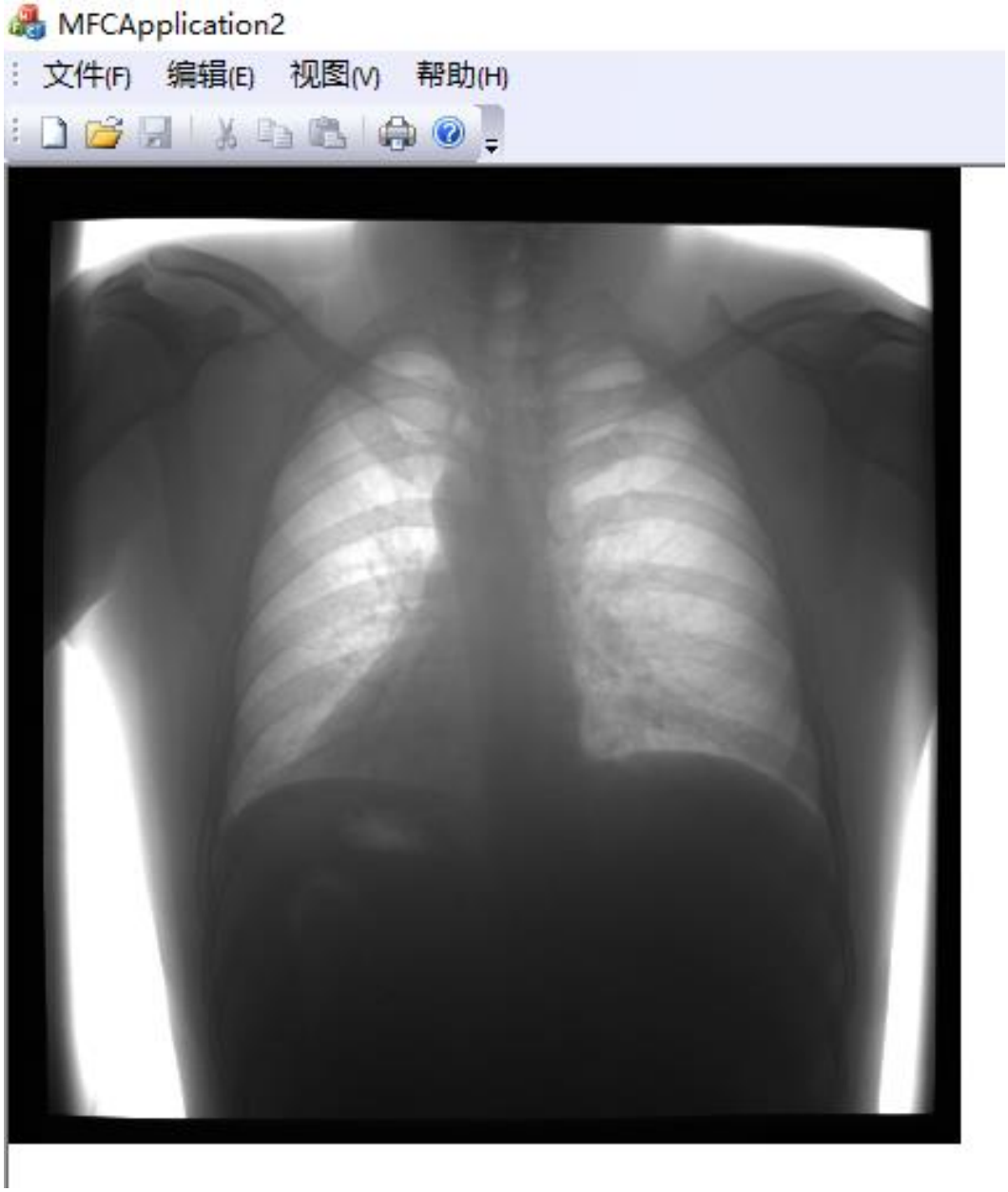
```
if(div<1)//放大操作，进行最邻近元插值
{
if(div==0)div=0.1;//检测放大值不为无穷大
    int t_width=width/div;//放大后的宽度
    int t_height=height/div;//放大后的高度
    testimg.Destroy();
    testimg=CImage();
    testimg.Create(t_width,t_height,24);
    float step=1.f/div;//放大后相对于原图而言每个像素之间填充的新的像素数量
    int i0=height/4;
    float width=2048,height=2099;//原始宽度和高度
    for(int i=0;i<i0;i++)
    {
        for(int j=0;j<width;j++)
        {
            unsigned int num=buffer[int(i*width)+j];//获取最邻近元的像素值
            int t1=j*step-step/2,t2=j*step+step/2;//计算最邻近元的坐标区域范围
                for(int k=t1;k<t2;k++)
                {
                    int t3=i*step-step/2,t4=i*step+step*2;//计算最邻近元的坐标区域范围
                    for(int m=t3;m<t4;m++)
                    {
                        if(k<=0)k=0;
                        if(k>=t_width)k=t_width-1;
                        if(m<=0)m=0;
                        if(m>=t_height)m=t_height-1;//处理边缘像素不能内存越界
                        testimg.SetPixel(k,m,RGB(num,num,num));//设置最邻近元像素灰度值相同
                    }
                }
        }
    }
```

2.3 移动图像算法

```
void CMFCApplication2View::OnLButtonDown(UINT nFlags, CPoint point)
{
    // TODO: 在此添加消息处理程序代码和/或调用默认值

    //CView::OnLButtonDown(nFlags, point);
    pressed=true;
    lastPosX=point.x;
    lastPosY=point.y;

    //lastImgX
    Invalidate();
}


void CMFCApplication2View::OnLButtonUp(UINT nFlags, CPoint point)
{
    pressed=false;
    CView::OnLButtonUp(nFlags, point);
    lastImgX=lastImgX+point.x-lastPosX;
    lastImgY=lastImgY+point.y-lastPosY;//左侧按键按下的时候记录初始图像
                                       //坐标值和初始鼠标坐标值
}


void CMFCApplication2View::OnMouseMove(UINT nFlags, CPoint point)
{
    if(pressed==true)//检测到左键持续按住的时候进行图片拖动
    {
        posX=lastImgX+(point.x-lastPosX);
        posY=lastImgY+(point.y-lastPosY);//拖动位置即为鼠标相对于初始位置的位移
        Invalidate();
    }
}
```

3.程序测试结果和评价

3.1 读取三张不同图片，5 倍缩放

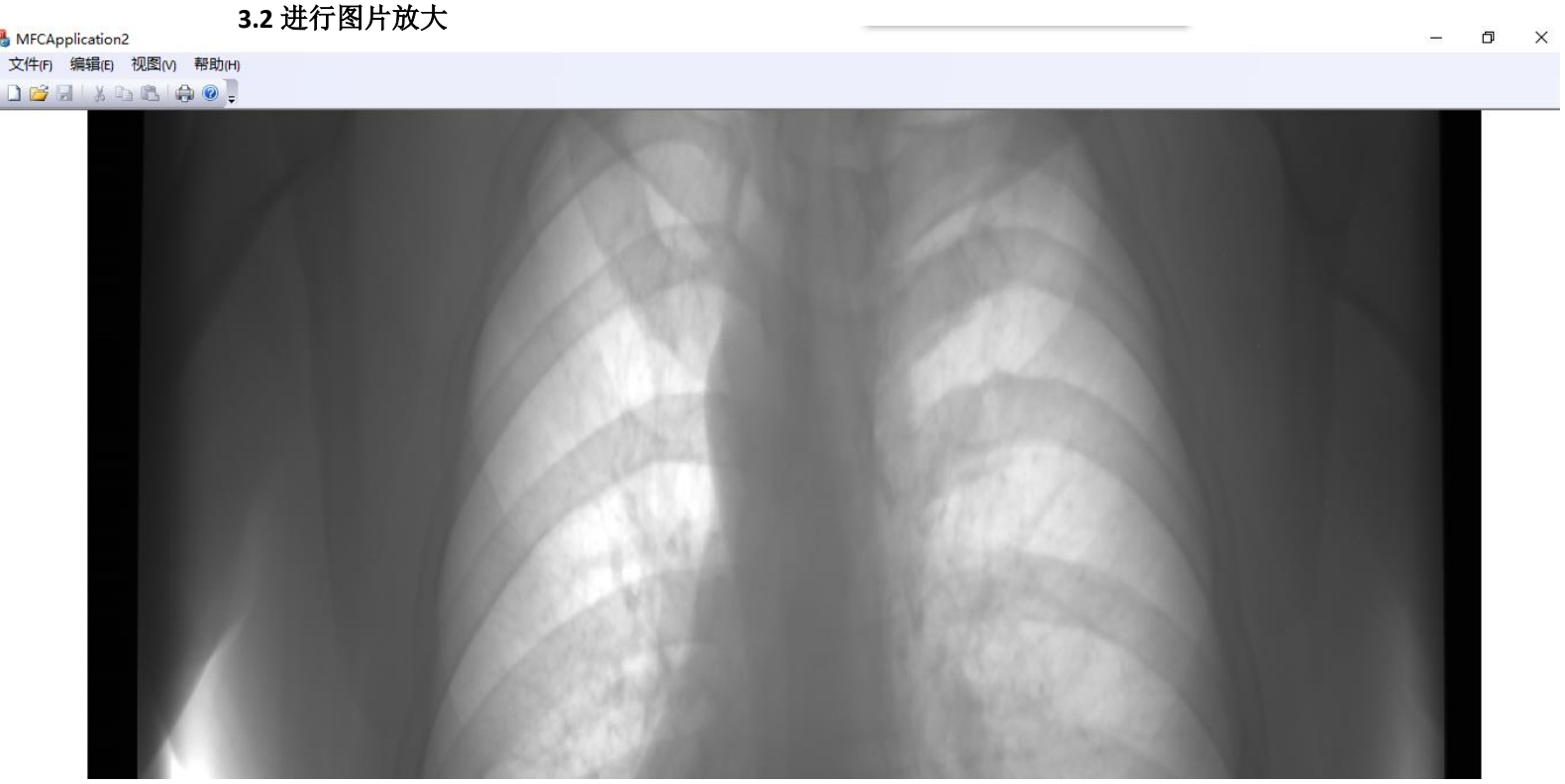


图一、肺叶

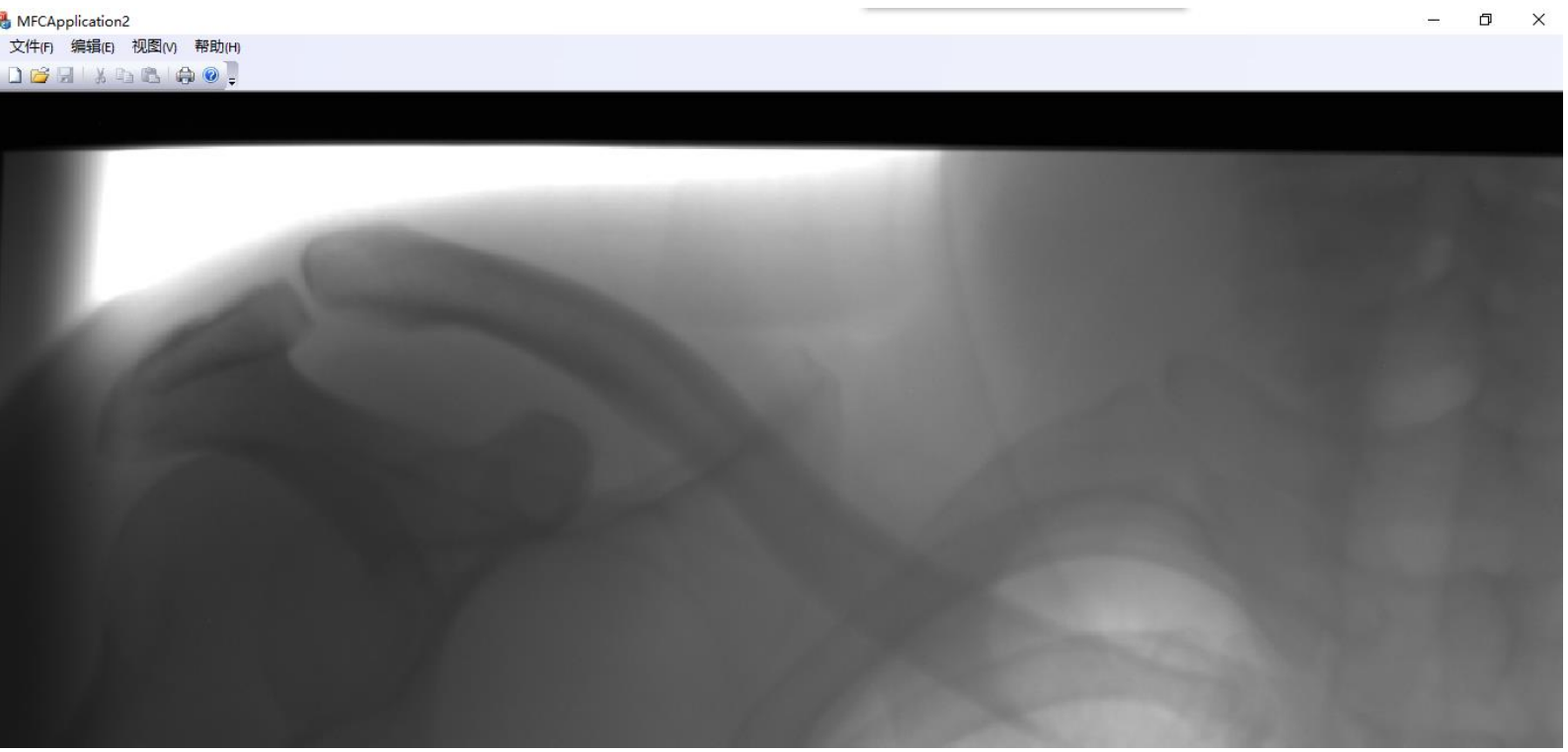
图二、脊椎

图三、髋

3.2 进行图片放大



图四、脊椎放大 1 倍



图五、脊椎放大 3 倍

3.3 程序运行评价

程序基本能够实现需求目标，可以对图片进行读取并显示，可以进行拖拽，可以进行放大和缩小。缺点在于进行放大操作的时候，当图片达到一定尺寸的时候，程序运行时间会大大加长。针对这个问题，主要是需要考虑调整放大图像算法，同时考虑对像素的访问不要使用 SetPixel 函数而是直接用指针进行数据操作。