# Homework3 作业报告

邱荻 2000012852 信息科学技术学院

#基础型

#（基础型，编程工具只限 Matlab）
模拟 Matlab 的 imresize()写一个你自己的 imresize()函数，至少应实现‘nearest’和‘bilinear’两种方法。附件的“football.jpg”和“kids.tiff”可作测试之用。

写 my_imresize 来处理两种不同插值的情况

```
function resized_image = my_imresize(image, scale, method)
    if nargin < 3
        method = 'bilinear'; % 没有第三个参数的话 就默认使用双线性插值
    end

    if strcmp(method, 'nearest')
        resized_image = nearest_interpolation(image, scale);
    end
    if strcmp(method, 'bilinear')
        resized_image = bilinear_interpolation(image, scale);
    end
end
```

## 对于 nearest

首先确定新图像的宽和高，以及两个方向的缩放比例
然后根据原理计算出每个像素的值

### 原理

最近邻插值，也称为零阶插值。计算原理为，目标图像位置直接采用与它最邻近位置的原始图像的像素点为其赋值。

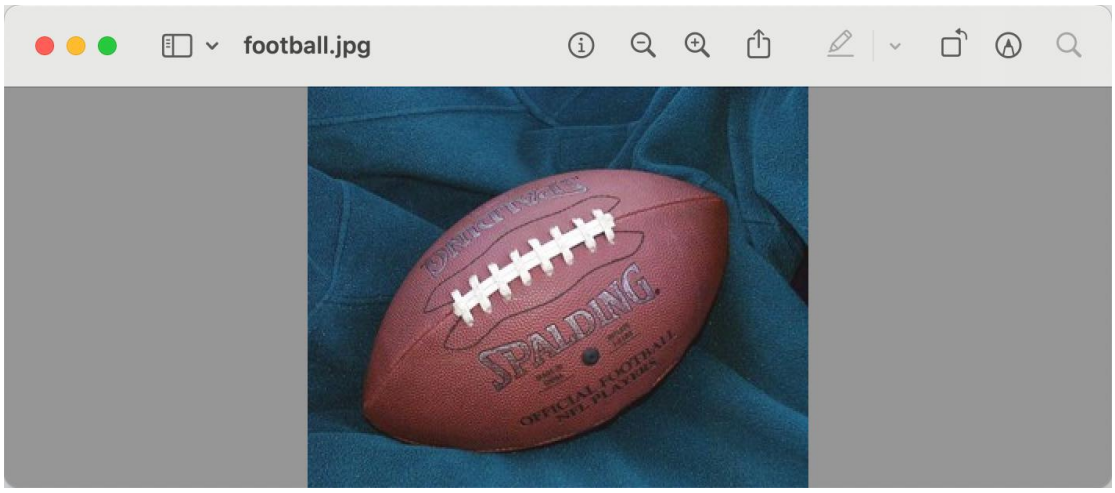
目标图像位置 `(dst_x, dst_y)` 最邻近的原始图像位置 `(src_x, src_y)` 的计算：`src_x = int(dst_x / scale_x)`；`src_y = int(dst_y / scale_y)`，其中 `scale_x` 和 `scale_y` 分别表示在图像宽度方向和高度方向的缩放比例。

### 代码

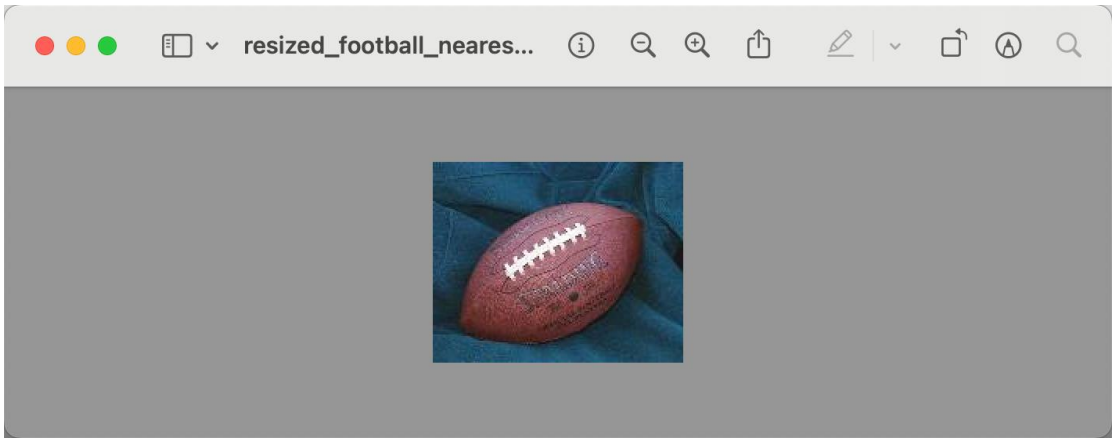
```
function resized_image = nearest_interpolation(image, scale)
    if numel(scale) == 1
        new_height = round(size(image, 1) * scale);
        new_width = round(size(image, 2) * scale);
        scale_x = scale;
        scale_y = scale;
    else
        new_height = scale(1);
        new_width = scale(2);
        scale_x = new_height/size(image, 1);
        scale_y = new_width/size(image, 2);
    end
    resized_image = zeros(new_height, new_width, size(image, 3), class(image));
    for i = 1:new_height
        for j = 1:new_width
            src_i = min(round(i/scale_x), size(image, 1));
            src_j = min(round(j/scale_y), size(image, 2));
            resized_image(i, j, :) = image(src_i, src_j, :);
        end
    end
end
```

**结果：**

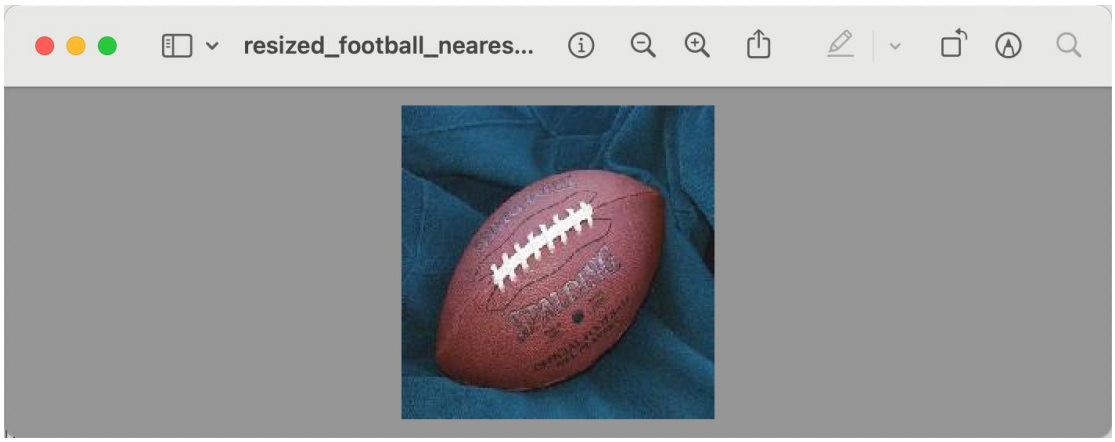
原图




以 scaleFactor = 0.5 nearest 为参数 resize 之后




以 scaleFactor = [200,200] nearest 为参数 resize 之后



# 对于 bilinear

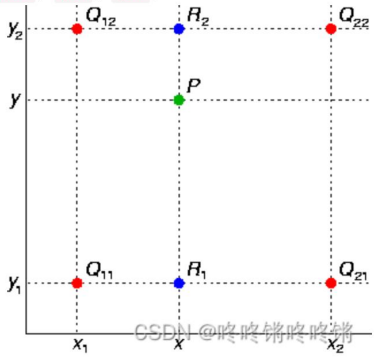
## 原理

双线性插值，又称双线性内插。其核心思想是在x和y两个方向分别进行线性插值。

1） 将目标图像的位置 `(dst_x, dst_y)` 映射到原图 `P(src_x, src_y)` 中，但这时取得的是 `float` 格式的结果；

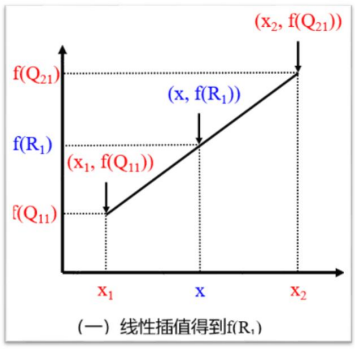
```
src_x = float(dst_x / scale_x)
src_y = float(dst_y / scale_y)
```

2）得到 `P` 点在原图中最邻近的4个点 `Q11`、`Q12`、`Q21`、`Q22`；

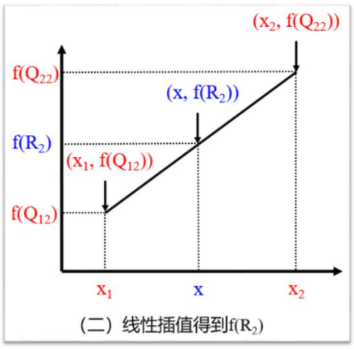




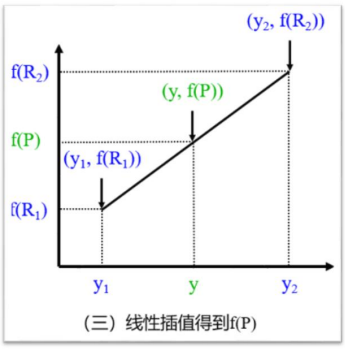
3）首先在x方向进行两次线性插值得到 `R1` 和 `R2` 两个点的像素值 `f(R1)` 和 `f(R2)`，然后再在y方向一次线性插值得到最终点P的像素值 `f(P)`，将该值赋值给目标图像 `(dst_x, dst_y)` 【注意此处如果先在y方向插值，再在x方向插值，其结果是一样的】



（一）线性插值得到f(R1)



（二）线性插值得到f(R2)



（三）线性插值得到f(P)



**计算公式：**

$$f(x, y_1) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21}),$$

$$f(x, y_2) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22}).$$

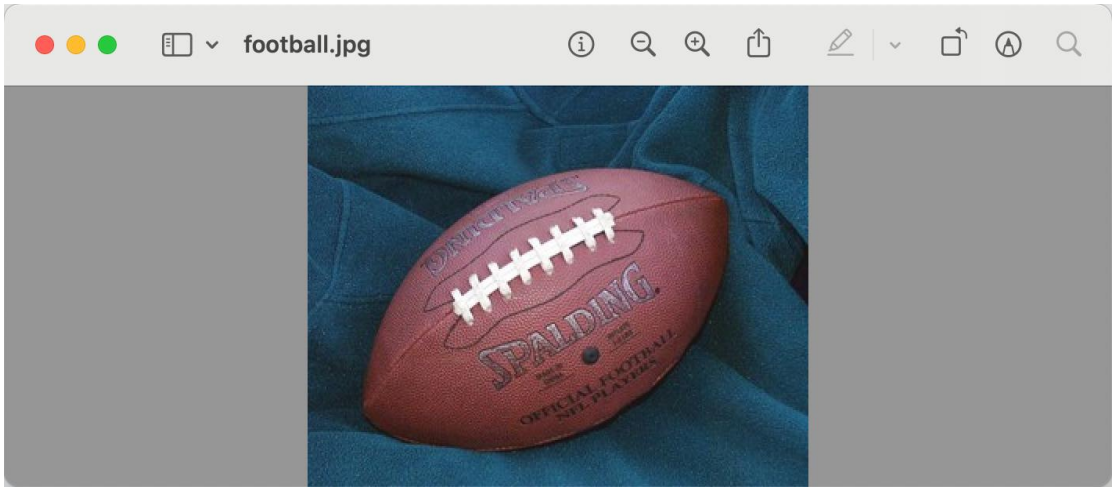
$$\begin{aligned} f(x, y) &\approx \frac{y_2 - y}{y_2 - y_1} f(x, y_1) + \frac{y - y_1}{y_2 - y_1} f(x, y_2) \\ &= \frac{y_2 - y}{y_2 - y_1} \left( \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21}) \right) + \frac{y - y_1}{y_2 - y_1} \left( \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22}) \right) \\ &= \frac{1}{(x_2 - x_1)(y_2 - y_1)} \big( f(Q_{11})(x_2 - x)(y_2 - y) + f(Q_{21})(x - x_1)(y_2 - y) + f(Q_{12})(x_2 - x)(y - y_1) + f(Q_{22})(x - x_1)(y - y_1) \big) \\ &= \frac{1}{(x_2 - x_1)(y_2 - y_1)} \begin{bmatrix} x_2 - x & x - x_1 \end{bmatrix} \begin{bmatrix} f(Q_{11}) & f(Q_{12}) \\ f(Q_{21}) & f(Q_{22}) \end{bmatrix} \begin{bmatrix} y_2 - y \\ y - y_1 \end{bmatrix}. \end{aligned}$$

## 代码

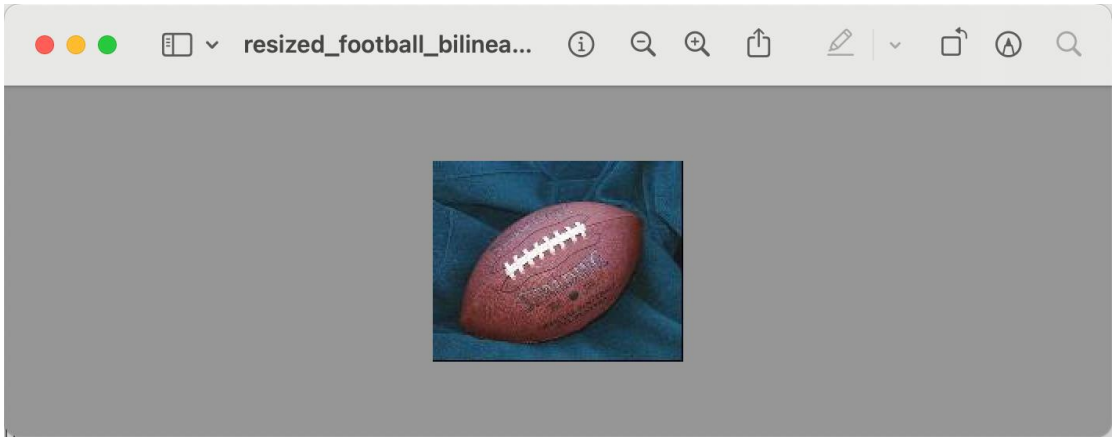
```
function resized_image = bilinear_interpolation(image, scale)
   if numel(scale) == 1
        new_height = round(size(image, 1) * scale);
        new_width = round(size(image, 2) * scale);
        scale_x = scale;
        scale_y = scale;
   else
        new_height = scale(1);
        new_width = scale(2);
        scale_y = new_height/size(image, 1);
        scale_x = new_width/size(image, 2);
    end
    resized_image = zeros(new_height, new_width, size(image, 3), class(image));
    for i = 1:new_height
        for j = 1:new_width
            x = j/scale_x;
            y = i/scale_y;
            x1 = floor(x);
            y1 = floor(y);
            x2 = min(x1+1, size(image, 2)-1);
            y2 = min(y1+1, size(image, 1)-1);
            Q11 = image(y1, x1, :);
            Q21 = image(y1, x2, :);
            Q12 = image(y2, x1, :);
            Q22 = image(y2, x2, :);
            if x2 == x1
                R1 = Q11;
                R2 = Q12;
            else
                R1 = Q11 * (x2 - x)/(x2 - x1) + Q21 * (x - x1)/(x2 - x1);
                R2 = Q12 * (x2 - x)/(x2 - x1) + Q22 * (x - x1)/(x2 - x1);
            end
            if y2 == y1
                P = R1;
            else
                P = R1 * (y2 - y)/(y2 - y1) + R2 * (y - y1)/(y2 - y1);
            end
            resized_image(i, j, :) = P;
        end
    end
end
```

## 结果:

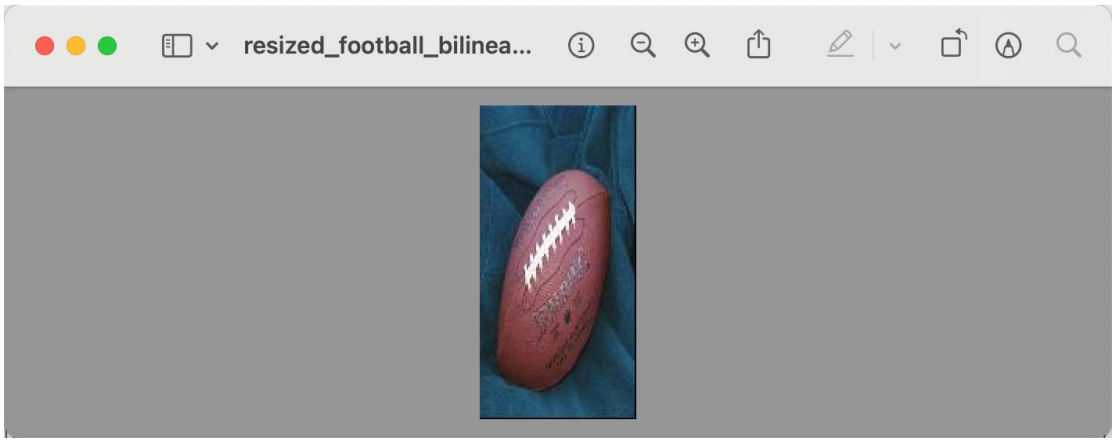
原图



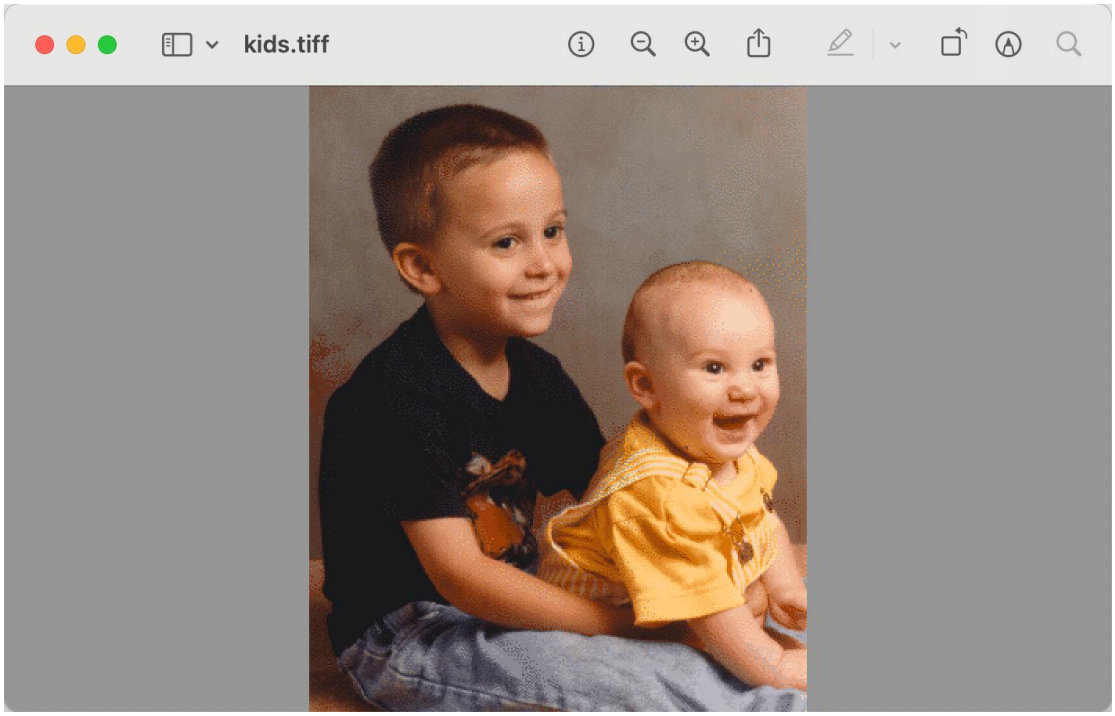
以 scaleFactor = 0.5, bilinear 为参数 resize 后



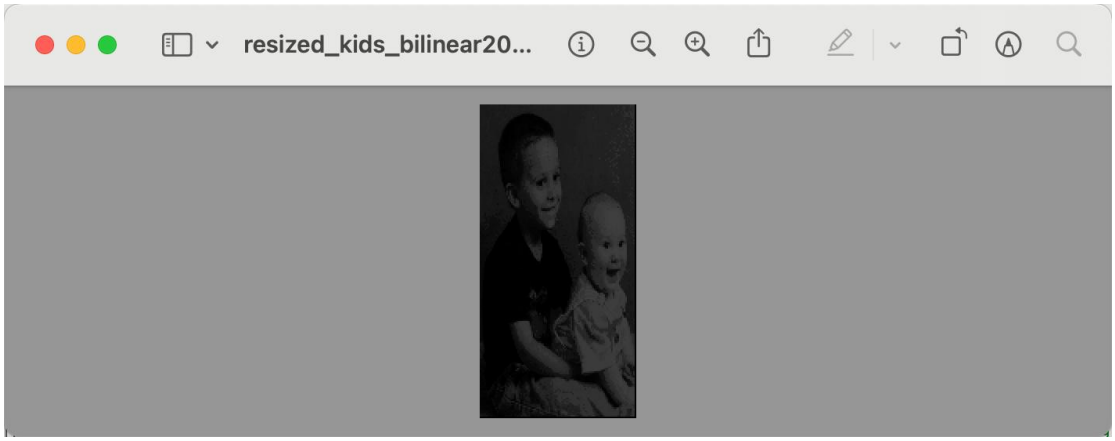

以 scaleFactor = [200,100], bilinear 为参数 resize 后




原图



以 [200,100] bilinear 为参数 resize 之后




以 0.5 bilinear 为参数 resize 之后