西安交通大学

计算机视觉与模式识别

计算机 53 班

龙思宇

2150500103

1. 实现一个函数，edge = non_maximum_suppression(magnitude, angle, edge); 实现最大值抑制的功能；

   源代码

```
function edge = non_maximum_suppression(magnitude, angle, edge)
    [nr,nc] = size(edge);
    for y = 2 : (nr -1)
        for x = 2 : (nc - 1)
            switch angle(y,x)
                case 0
                  if magnitude(y,x) > magnitude(y,x - 1) &&
                  magnitude(y,x) > magnitude(y,x + 1)
                      edge(y,x) = 1;
                   end
                case pi/4
                  if magnitude(y,x) > magnitude(y + 1,x - 1)
                  && magnitude(y,x) > magnitude(y - 1,x + 1)
                      edge(y,x) = 1;
                   end
                case pi/2
                  if magnitude(y,x) > magnitude(y + 1,x) &&
                  magnitude(y,x) > magnitude(y - 1,x)
                      edge(y,x) = 1;
                   end
                case 3*pi/4
                  if magnitude(y,x) > magnitude(y - 1,x - 1)
                  && magnitude(y,x) > magnitude(y + 1,x + 1)
                      edge(y,x) = 1;
                   end
                end
            end
        end
    end
```

2. 实现这一个函数，linked_edge = hysteresis_thresholding(threshold_low, threshold_high, linked_edge, edge);实现迟滞的边缘链接功能；

   源代码

```
function linked_edge = hysteresis_thresholding(threshold_low,
threshold_high, linked_edge, edge)
    set(0,'RecursionLimit',10000);
    [nr,nc] = size(edge);
    for y = 2 : (nr -1)
        for x = 2 : (nc - 1)
            if edge(y,x) > threshold_high && linked_edge(y,x) ~=
            1
                linked_edge(y,x) = 1;
                linked_edge =
                connect(threshold_low,linked_edge,edge,y,x);
            end
        end
    end
end


function linked_edge = connect(threshold_low,
linked_edge,edge,y,x)
    neighbour=[-1 -1;-1 0;-1 1;0 -1;0 1;1 -1;1 0;1 1];
    [m,n] = size(edge);
    for k = 1:8
        yy = y + neighbour(k,1);
        xx = x + neighbour(k,2);
        if yy > 1 && yy <= m && xx > 1 && xx <=n
            if edge(yy,xx) > threshold_low && linked_edge(yy,xx)
            ~= 1
                linked_edge(yy,xx) = 1;
                linked_edge = connect(threshold_low, linked_edge,
                edge,yy,xx);
            end
        end
    end
end
```

3. 找自己拍摄一组图像，用自己编写的 Canny 边缘检测算法去计算边缘点；

   原图

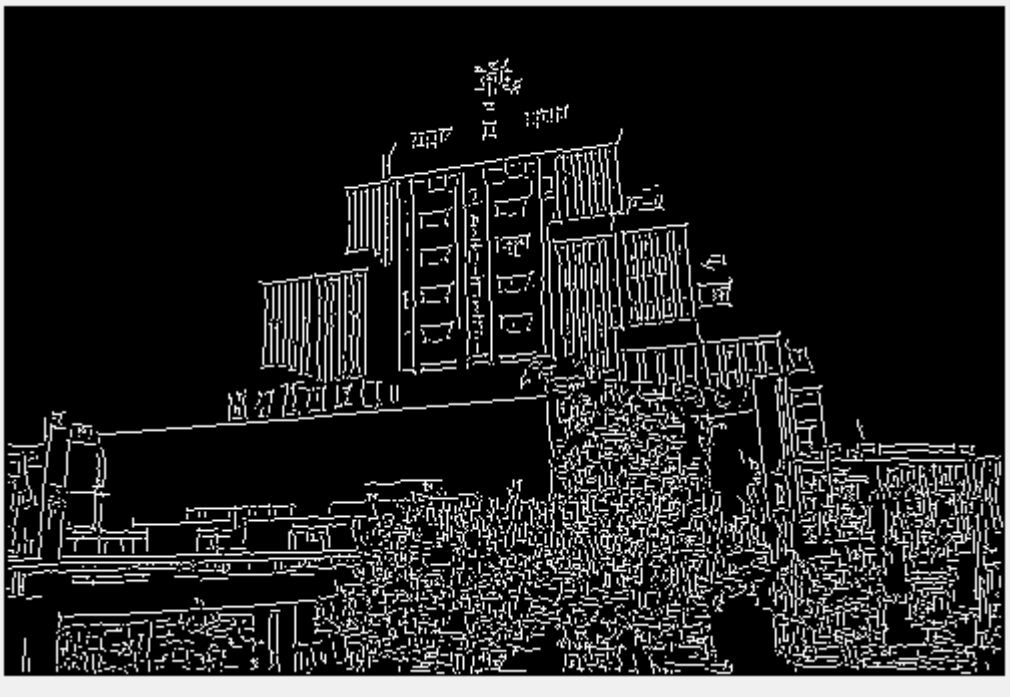
最大值抑制后图像

最终结果

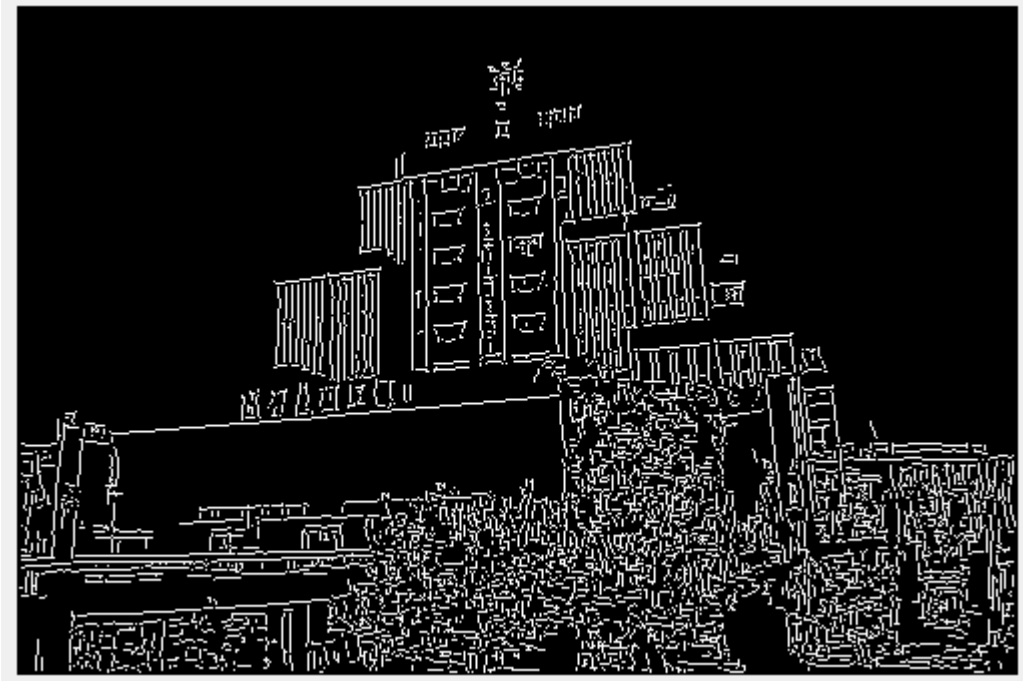
4. 分析 threshold_low 和 threshold_high 对边缘生成的影响，尝试结合实验结果从理论上分析这两者的影响；

   当 threshold_high 固定为 0.175 时，逐渐提高 threshold_low

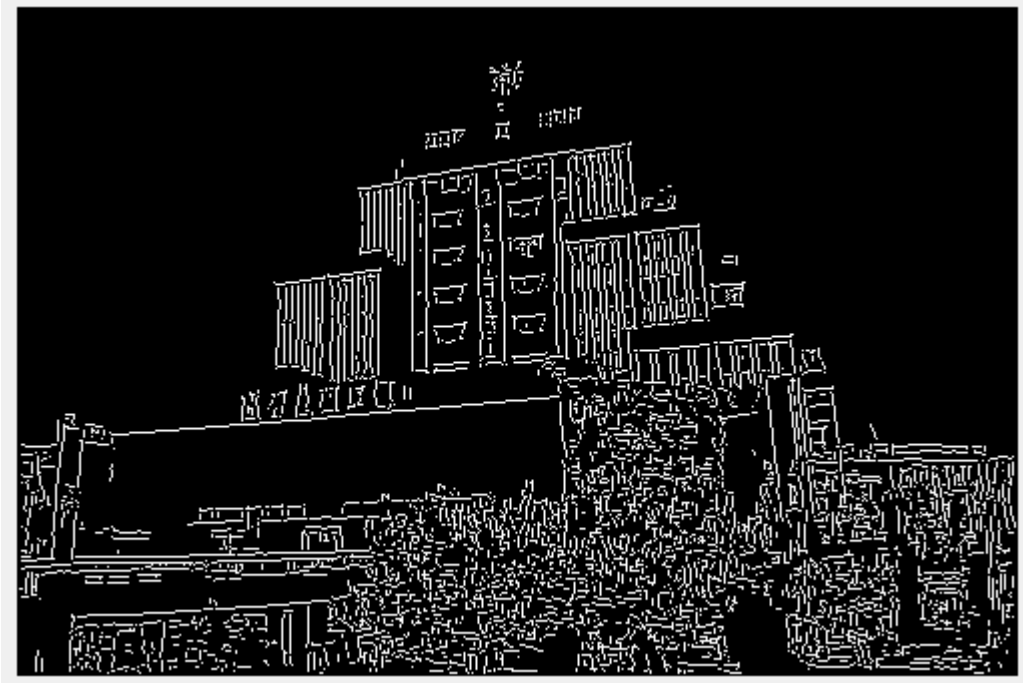
   threshold_low = 0.050



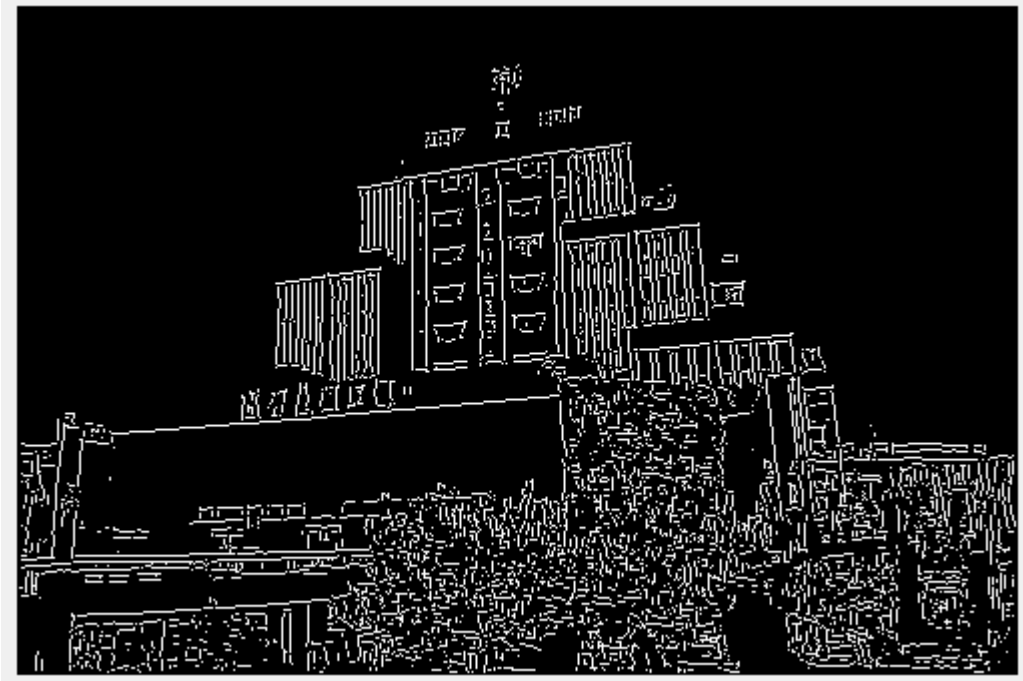
   threshold_low = 0.100



threshold_low = 0.150



threshold_low = 0.175



当 threshold_low 固定为 0.035 时，逐渐降低 threshold_high

threshold_high = 0.150

threshold_high = 0.100

threshold_high = 0.050

threshold_high = 0.035

从上面的例子可以看到当 threshold_low 变大的过程中，细节是逐渐变少的，这是因为提高了成为弱边界的条件，使得可能是边界的点变少了，原来可能是边界的点在 threshold_low 提高后变成了非边界点，而当 threshold_high 变小的过程中，细节是逐渐变多的，这是因为降低了成为强边界的条件，使得原来只是可能是边界的点成为了强边界点。

5. 结合实验结果分析最大值抑制对边缘检测的影响；

   当未做最大值抑制时，所得图像

做了最大值抑制后的图像

明显前者比后者线条粗了很多，多出了许多和我们判断边界无用的点，大大增加了计算量。

6. 结合实验结果分析迟滞的边缘链接对边缘检测结果的影响；

4 中当 threshold_low = 0.175 和 threshold_high = 0.035 时，结果是

一个二值的边缘链接，这两张图片一个丢失了太多细节，可能抹去了本该是边界的点，一个细节又太多，可能使过多本来不是边界的地方成为边界，使用迟滞链接，一来保存了高幅值的一定是边界的部分，二来抑制了幅值过小的非边界的部分，最后还通过链接的方式一定程度上避免了弱边界丢失的问题。