機器視覺作業報告

影像細線化

姓名:李珮慈
學號:00657124
日期:2020/06/03

● 方法

此為對圖片做二值化。

先選定一個常數 C，然後對圖片做捲積，比 C 大的設為 1，比 C 小的設為 0。

```
def __call__(self,inputs):
    x = inputs-tf.nn.conv2d(inputs,self.filters,strides=(1,1),padding='SAME')
    x = tf.where(x>=self.C,tf.constant([[[1.]]],dtype=tf.float32),tf.constant([[[0.]]],dtype=tf.float32))
    return x
```

此為實現細線化的兩個規則計算。

```
@staticmethod
def _surface_patterns():
    pattern1 = []
    pattern2 = []
    image   = np.zeros((9,))
    image[4]= 1
    clockwise=[(0,0),(0,1),(0,2),(1,2),(2,2),(2,1),(2,0),(1,0),(0,0)]
    for i in range(256):
        n = 0
        for j in range(4):
            image[j] = 2*(i//(2**j) % 2)-1
            if image[j] == 1:
                n += 1
        for j in range(4,8):
            image[j+1] = 2*(i//(2**j) % 2)-1
            if image[j+1] == 1:
                n += 1
        if n >= 2 and n <= 6:
            a = np.reshape(image,(3,3))
            sp = 0
            for j in range(8):
                if a[clockwise[j][0],clockwise[j][1]]==-1 and a[clockwise[j+1][0],clockwise[j+1][1]]==1:
                    sp += 1
            if sp == 1:
                if (a[0,1]==-1 or a[1,2]==-1 or a[2,1]==-1) and (a[1,2]==-1 or a[2,1]==-1 or a[1,0]==-1):
                    pattern1.append(np.reshape(a.copy(),(3,3,1,1)))

                if (a[0,1]==-1 or a[1,0]==-1 or a[1,2]==-1) and (a[0,1]==-1 or a[2,1]==-1 or a[1,0]==-1):
                    pattern2.append(np.reshape(a.copy(),(3,3,1,1)))
    return tf.constant(np.concatenate(pattern1,axis=-1),dtype=tf.float32),tf.constant(np.concatenate(pattern2,axis=-1),
                                                                                      dtype=tf.float32)
```

將計算完的圖做點的刪除(符合規則就刪掉，直到陣列裡面沒有 0 為止)

Rule1:

掃描圖片，再計算張量中各維度的最大元素。

若 z=9，就設為 1，否就為 0。(意為刪除該點)

若 z=9，就設為 1，否就設為 x。

Rule2:

掃描圖片，再計算張量中各維度的最大元素。

若 z=9，就設為 1，否就為 0。(意為刪除該點)

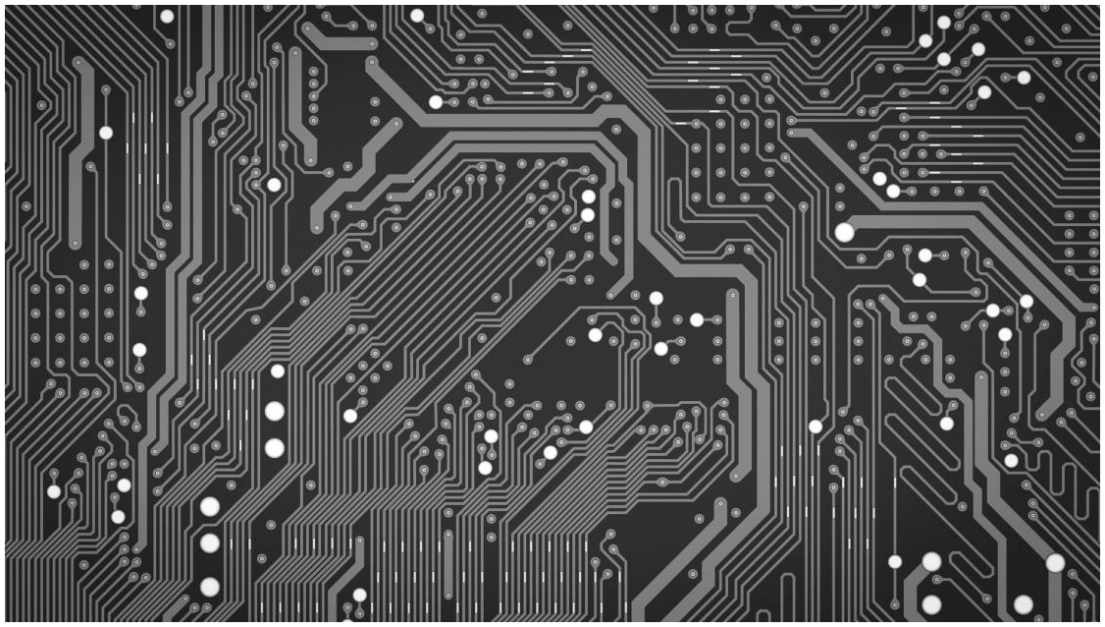
若 z=9，就設為 1，否就設為 x。

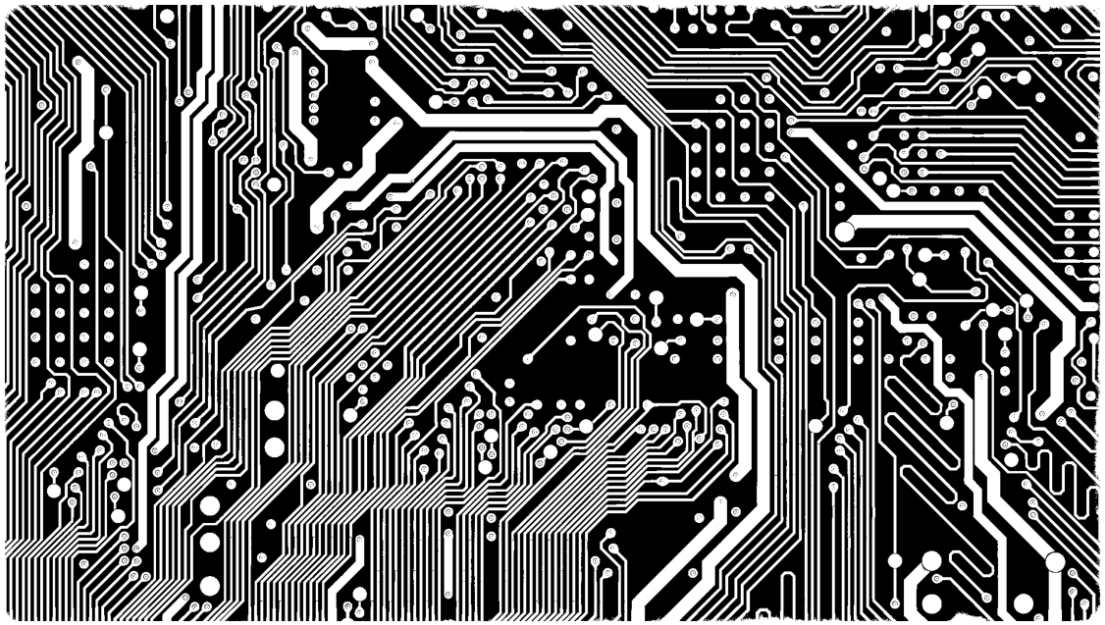
檢查 c1 和 c2 的值，如果都為 0 就表示不會在 1 和-1 之間轉換，跳出迴圈。

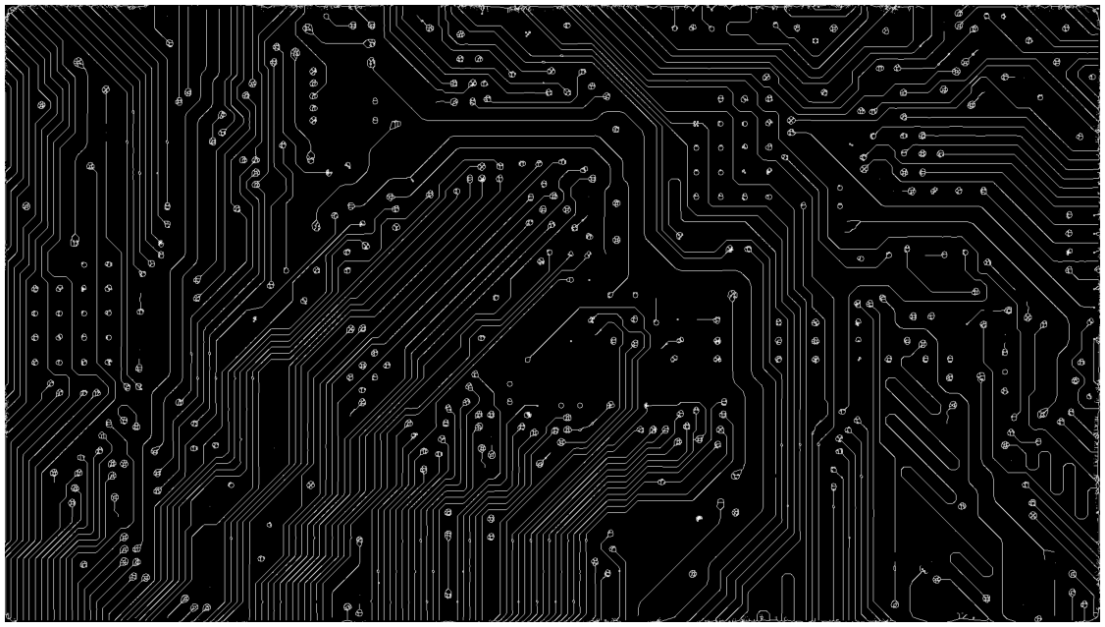
```
def __call__(self,inputs):
    #  do thinning
    #  padding is required
    x = tf.pad(inputs,tf.constant([[0,0],[1,1],[1,1],[0,0]]),constant_values=-1.0)
    while True:
        #rule 1
        z =tf.nn.conv2d(x,self.filters1,strides=(1,1),padding='SAME')
        z =tf.math.reduce_max(z,axis=-1,keepdims=True)
        c1=tf.where(z==9,tf.constant([[[1.]]],dtype=tf.float32),tf.constant([[[0.]]],dtype=tf.float32))
        x =tf.where(z==9,tf.constant([[[-1.]]],dtype=tf.float32),x)

        #rule 2
        z =tf.nn.conv2d(x,self.filters2,strides=(1,1),padding='SAME')
        z =tf.math.reduce_max(z,axis=-1,keepdims=True)
        c2=tf.where(z==9,tf.constant([[[1.]]],dtype=tf.float32),tf.constant([[[0.]]],dtype=tf.float32))
        x =tf.where(z==9,tf.constant([[[-1.]]],dtype=tf.float32),x)
        if tf.reduce_sum(c1) == 0 and tf.reduce_sum(c2) == 0:
            break
    outputs = x[:,1:-1,1:-1,:]
    return outputs
```

● 結果

Github: https://github.com/muyumiya1201/00657124_MV_HW3







● 結論

剛開始理解 thinning 那兩個 rule 不太容易，後來找了多方的網站解釋，慢慢了解它的規則。以某點為中心去找身邊 8 格的值，符合的值就將其刪除(程式中將其值設為0)。後來發現，其實只要中心點的上下左右其中有一個為0，基本上就符合刪除的規定

- **參考**

  https://blog.csdn.net/weixin_40977054/article/details/96888371

  https://rocky69.pixnet.net/blog/post/218271623-%5B%E8%BD%89%5D%E5%9C%96%E5%83%8F%E7%B4%B0%E5%8C%96%EF%BC%88%E9%AA%A8%E6%9E%B6%E5%8C%96%EF%BC%89%E7%AE%97%E6%B3%95-%E5%88%86%E6%9E%90

  https://blog.csdn.net/weixin_38419133/article/details/98499664