第九章作业 王晨曦

Created @April 20, 2022 10:58 AM

姓名：王晨曦

班级：计算机96

学号：2196123413

得分：

---

1. **第一题**

- 读入数据

In [2]:
```
import numpy as np
data1 = np.zeros((5, 5))
data2 = np.zeros((5, 5))
for i in range(5):
    for j in range(5):
        data1[i][j]=i*5+j+1
        data2[i][j]=10+i*5+j+1
print(data1)
print(data2)
```

```
[[ 1.  2.  3.  4.  5.]
 [ 6.  7.  8.  9. 10.]
 [11. 12. 13. 14. 15.]
 [16. 17. 18. 19. 20.]
 [21. 22. 23. 24. 25.]]
[[11. 12. 13. 14. 15.]
 [16. 17. 18. 19. 20.]
 [21. 22. 23. 24. 25.]
 [26. 27. 28. 29. 30.]
 [31. 32. 33. 34. 35.]]
```

- 写计算SAD误差的函数，非常简单，对应相减求绝对值，然后整个矩阵求和即可

```
[9]: def SAD(data1,data2):
         result = []
         for i in range(5):
             for j in range(5):
                 result.append(np.abs(data1[i][j]-data2[i][j]))
         return sum(result)
```

- 写计算MSE误差的函数，对应相减求平方，然后整个矩阵求和，并除以矩阵元素数量得平均即可

```
[17]: def MSE(data1,data2):
          result = []
          for i in range(5):
              for j in range(5):
                  result.append((data1[i][j]-data2[i][j])*(data1[i][j]-data2[i][j]))
          result = sum(result)
          result = result/25
          return result
```

- 写计算NCC误差的函数，先用numpy库函数np.mean( )和np.std( )求矩阵的均值和标准差（要注意numpy.std() 求标准差的时候默认是除以 n 的，即是有偏的，np.std无偏样本标准差方式为加入参数 ddof = 1），接下来部分按照公式计算即可

```
]: def NCC(data1,data2):
       result=[]
       u1 = np.mean(data1)
       u2 = np.mean(data2)
       std1 = np.std(data1,ddof=1)
       std2 = np.std(data2,ddof=1)#numpy.std() 求标准差的时候默认是除以
       for i in range(5):
           for j in range(5):
               result.append((data1[i][j]-u1)*(data2[i][j]-u2))
       result = sum(result)
       return result/(24*std1*std2)
```

- **SAD、MSE、NCC计算结果分别为250、100、1**

```
]: sad = SAD(data1, data2)
   mse = MSE(data1, data2)
   ncc = NCC(data1, data2)
   print(sad)
   print(mse)
   print(ncc)
```

```
250.0
100.0
1.0
```

2. 第二题

2、 给定一幅5 × 6的图像如下所示：

0	0	0	0	0	0
0	0	1	1	0	0
0	0	1	1	0	0
0	0	1	1	0	0
0	0	0	0	0	0

（1）计算 Ix，Iy

（2）利用如下的平滑模板

$$w(m,n) = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

计算平滑后的 Ixx,Iyy 和 Ixy

（4） 计算每一点的 corner ness，使用k = 0.04

- 读入数据

[27]:
```
data3 = np.zeros((7,8))
for i in range(7):
    for j in range(8):
        if i == 2 or i == 3 or i ==4:
            if j == 3 or j ==4:
                data3[i][j]=1
print(data3)
```

```
[[0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 1. 0. 0. 0.]
 [0. 0. 0. 1. 1. 0. 0. 0.]
 [0. 0. 0. 1. 1. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0.]]
```

- 写计算Ix和Iy的函数，并计算Ix和Iy
  - 函数

[28]:
```
def Ix(data3):
    result =  np.zeros((5,6))
    for i in range(1,6):
        for j in range(1,7):
            result[i-1][j-1]=data3[i][j-1]-data3[i][j+1]
    return result
```

[30]:
```
def Iy(data3):
    result =  np.zeros((5,6))
    for i in range(1,6):
        for j in range(1,7):
            result[i-1][j-1]=data3[i-1][j]-data3[i+1][j]
    return result
```

  - **Ix和Iy计算结果**

[69]:
```
ix = Ix(data3)
iy = Iy(data3)
print(ix)
print()
print(iy)
```

```
[[ 0.  0.  0.  0.  0.  0.]
 [ 0. -1. -1.  1.  1.  0.]
 [ 0. -1. -1.  1.  1.  0.]
 [ 0. -1. -1.  1.  1.  0.]
 [ 0.  0.  0.  0.  0.  0.]]

[[ 0.  0. -1. -1.  0.  0.]
 [ 0.  0. -1. -1.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.]
 [ 0.  0.  1.  1.  0.  0.]
 [ 0.  0.  1.  1.  0.  0.]]
```

- 分别令x自身元素对应相乘、y自身元素对应相乘、x和y对应元素相乘，得到中间矩阵，由于模板运算前后要得到相同面积，所以需要对中间矩阵四周进行值为0的填充，接下来对中间矩阵使用模板进行平滑，得到Ixx、Iyy和Ixy
  - 中间矩阵

```
[[0. 0. 0. 0. 0. 0.]
 [0. 1. 1. 1. 1. 0.]
 [0. 1. 1. 1. 1. 0.]
 [0. 1. 1. 1. 1. 0.]
 [0. 0. 0. 0. 0. 0.]]

[[0. 0. 1. 1. 0. 0.]
 [0. 0. 1. 1. 0. 0.]
 [0. 0. 0. 0. 0. 0.]
 [0. 0. 1. 1. 0. 0.]
 [0. 0. 1. 1. 0. 0.]]

[[ 0.  0. -0. -0.  0.  0.]
 [ 0. -0.  1. -1.  0.  0.]
 [ 0. -0. -0.  0.  0.  0.]
 [ 0. -0. -1.  1.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.]]
```

  - 填充边缘后

```
[[0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 1. 1. 1. 1. 0. 0.]
 [0. 0. 1. 1. 1. 1. 0. 0.]
 [0. 0. 1. 1. 1. 1. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0.]]
[[0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 1. 0. 0. 0.]
 [0. 0. 0. 1. 1. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 1. 0. 0. 0.]
 [0. 0. 0. 1. 1. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0.]]
[[ 0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0. -0. -0.  0.  0.  0.]
 [ 0.  0. -0.  1. -1.  0.  0.  0.]
 [ 0.  0. -0. -0.  0.  0.  0.  0.]
 [ 0.  0. -0. -1.  1.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.]]
```

- 平滑模板函数

[51]:
```python
def moban(matrix,x,y):
    sum = 0
    for i in range(-1,2):
        for j in range(-1,2):
            sum = sum+matrix[x+i][y+j]
    return sum
def smooth(matrix):
    result = np.zeros((5,6))
    for i in range(1,6):
        for j in range(1,7):
            result[i-1][j-1]=moban(matrix,i,j)
    return result
```

- **平滑后的Ixx、Iyy和Ixy计算结果**

```
[[1. 2. 3. 3. 2. 1.]
 [2. 4. 6. 6. 4. 2.]
 [3. 6. 9. 9. 6. 3.]
 [2. 4. 6. 6. 4. 2.]
 [1. 2. 3. 3. 2. 1.]]

[[0. 2. 4. 4. 2. 0.]
 [0. 2. 4. 4. 2. 0.]
 [0. 2. 4. 4. 2. 0.]
 [0. 2. 4. 4. 2. 0.]
 [0. 2. 4. 4. 2. 0.]]

[[ 0.  1.  0.  0. -1.  0.]
 [ 0.  1.  0.  0. -1.  0.]
 [ 0.  0.  0.  0.  0.  0.]
 [ 0. -1.  0.  0.  1.  0.]
 [ 0. -1.  0.  0.  1.  0.]]
```

- 对每一个像素点（x，y），均从Ixx、Iyy和Ixy中取到对应值，然后临时构建一个2*2的海塞矩阵，使用numpy库函数对这个临时矩阵求特征值，然后按照R的公式求得（x，y）的corner ness值。
  - 写一个构建临时矩阵并对该矩阵求特征值的函数，返回值为特征值

```
]6]: def tz(ixx,iyy,ixy):
         temp = np.array([[ixx,ixy],[ixy,iyy]])
         tz = np.linalg.eig(temp)
         return tz[0]
```

  - **对每一个像素点（x，y）使用上面的函数，并将所得特征值代入R的公式，得到每一个像素点的corner ness如下：**

```
'4]: ness=np.zeros((5,6))
     for i in range (5):
         for j in range(6):
             temp=tz(ixx_after[i,j],iyy_after[i,j],ixy_after[i,j])
             ness[i,j]=temp[0]*temp[1]-0.04*(temp[0]+temp[1])*(temp[0]+temp[1])
     print(ness)
```

```
[[-0.04  2.36 10.04 10.04  2.36 -0.04]
 [-0.16  5.56 20.    20.    5.56 -0.16]
 [-0.36  9.44 29.24 29.24  9.44 -0.36]
 [-0.16  5.56 20.    20.    5.56 -0.16]
 [-0.04  2.36 10.04 10.04  2.36 -0.04]]
```