画像処理の基礎

T 班 1610581 堀田 大地

2018/12/27

1　目的

画像処理の基礎実装を通して，画像処理について学んだ．

2　課題 1 画素値と画像の関係

画素値によって色がどのように変化したかを確認した．

2.1　方法

#1 の部分に [1,255] の値を代入して確認した．

ソースコード 1　kadai1.py

```
gazo = zeros((10,10))
for x in range(10):
    for y in range(10):
        gazo[y][x] = 255 #1
```

2.2　結果と考察

値が小さくなるにつれて黒に近づいた．これは，RGB 要素を加法混色によって混ぜ合わせて色を生成しているからである．

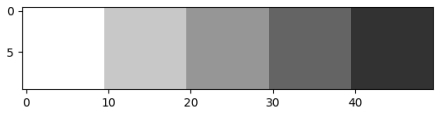

図 1　値と色の関係．右にいくにつれて値が小さくなっていっている．

3　課題 2 画素位置と画像の関係

配列のインデックスの数字と画像との対応を確認した．

3.1　方法

ソースコード 2　kadai2.py

```
gazo = zeros((10,10))
gazo[3][7] = 255
gazo_1[7][3] = 255
gazo_2[2][5] = 255
gazo_3[5][2] = 255
```

3.2　結果と考察

図 2 に上記のソースコードの結果を示す．1 次元目が y 軸，2 次元目が x 軸に対応していた．原点は左上で図形右側に x 軸，下側に y 軸の正の方向があった．

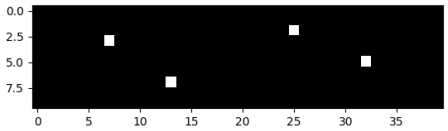

図 2　画像の任意座標に 255(白色) を代入した結果．左から gazo, gazo1, gazo2, gazo3 に対応する．

4　課題 3 図形の描画

輪郭が黒く，その内側の領域が白い正方形を生成した．

4.1　方法

ソースコード 3　kadai3.py

```
gazo = zeros((10,10))
for x in range(10):
    for y in range(10):
        if 0 < x < 9 and 0 < y < 9:
            gazo[y][x] = 255
```

4.2　結果と考察

結果を図 3 に示した．x，y 軸が [1,8] である場合に 255 を代入した．

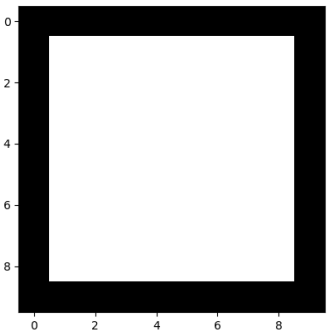

図 3　輪郭が黒くその内側の領域が白い正方形．

5　課題 4 フィルタ

移動平均フィルタ，ラプラシアンフィルタを実装した．

5.1　方法

移動平均，ラプラシアンフィルタの実装のコードを載せた．移動平均フィルタは自分の周囲 9 ピクセルの画素値の平均値を求めている．また，ラプラシアンフィルタは二次微分を利用して画像から輪郭を抽出している．

ソースコード 4　kadai4IdouHeikin.py

```
gazo_idou = zeros((12,12))
for x in range(1,11):
    for y in range(1,11):
        filter = [
            [0.5, 0.5, 0.5],
            [0.5, 0.5, 0.5],
            [0.5, 0.5, 0.5]
            ]
        gasochi = 0
        for xx in range(3):
            for yy in range(3):
                gasochi += gazo[y+yy
                    -1][x+xx-1] *
                    filter[yy][xx]
        gasochi = int(gasochi)
        if gasochi < 0:
            gasochi = 0
        elif gasochi > 255:
            gasochi = 255
        gazo_idou[y][x] = gasochi
```

ソースコード 5　kadai4Laplasian.py

```
gazo_lap = zeros((12,12))
for x in range(1,11):
    for y in range(1,11):
        filter = [
            [0.0, 1.0, 0.0],
            [1.0, -4.0, 1.0],
            [0.0, 1.0, 0.0]
            ]
        gasochi = 0
        for xx in range(3):
            for yy in range(3):
                gasochi += gazo[y+yy
                    -1][x+xx-1] *
                    filter[yy][xx]
        gasochi = int(gasochi)
        if gasochi < 0:
            gasochi = 0
        elif gasochi > 255:
            gasochi = 255
        gazo_lap[y][x] = gasochi
```

5.2　結果と考察

結果を図 4 に示す．結果よりそれぞれの役割がわかった．

1. 移動平均フィルタ
   注目領域周囲 9 ピクセルの平均値をそこに代入している．
2. ラプラシアンフィルタ
   カーネルの導出方法を示す．水平方向および垂直方向の画素値の一次微分は (1)，(2) で表せ

る．加えて，二次微分はもう一度差分を取ることで (3)，(4) で表せる．

$$I_x(x,y) = I(x+1,y) - I(x,y) \tag{1}$$
$$I_y(x,y) = I(x,y+1) - I(x,y) \tag{2}$$
$$\begin{aligned} I_{xx}(x,y) &= I(x+1,y) - I(x,y) - I(x,y) - I(x-1,y) \\ &= I(x-1,y) - 2I(x,y) + I(x+1,y) \end{aligned} \tag{3}$$
$$\begin{aligned} I_{yy}(x,y) &= I(x,y-1) - I(x,y) - I(x,y) - I(x,y+1) \\ &= I(x,y-1) - 2I(x,y) + I(x,y+1) \end{aligned} \tag{4}$$

よって，ラプラシアン $\nabla^2 I(x,y)$ は (5) で表せれる．

$$\begin{aligned} \nabla^2 I(x,y) &= I_{xx}(x,y) + I_{yy}(x,y) \\ &= I(x-1,y) + I(x,y-1) - 4I(x,y) \\ &\quad + I(x+1,y) + I(x,y+1) \end{aligned} \tag{5}$$

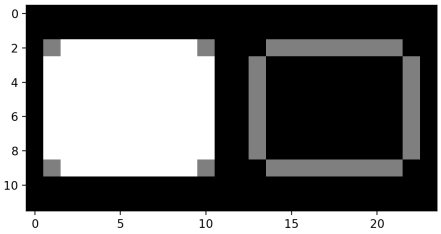

図 4　左:移動平均フィルタの結果，右:ラプラシアンフィルタの結果.

6　課題 5 ヒストグラムと二値化

画像の特徴を表わす二値画像が生成されるように作成した.

6.1　方法

ピクセル毎に，ある閾値より下回る画素値であれば，0 を代入，それ以外であれば 255 を代入した.

ソースコード 6　kadai5.py

```
def shori(gazo, gazo2, SIKII):
    """
    Args:
        gazo : 入力画像の配列.
        gazo2 : 出力画像の配列.
        SIKII : 閾値.
    """
    for x in range(gazo.shape[1]):
        for y in range(gazo.shape[0]):
            value = gazo[y][x]
            if value < SIKII:
                gazo2[y][x] = 0
            else:
                gazo2[y][x] = 255
    return gazo2
```

6.2　結果と考察

閾値が大きいと望んでいないノイズも含まれた.

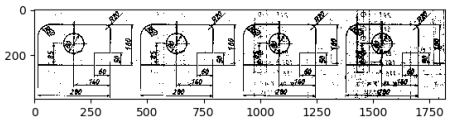

図 5　左から閾値が 220，225，230，235 の結果.

7　課題 6 ヒストグラムと階調変換

濃淡がよりはっきりとする画像を生成した.

7.1　手法

画像の正規化を行なった．また，平均輝度と輝度値の標準偏差は好みで決めた.

ソースコード 7　kadai5.py

```
def tra(img, mean_kido, std):
    """
    Args:
        img : 入力画像.
        mean_kido : 平均輝度.
        std : 輝度値の標準偏差.
    """
    img = (img - np.mean(img))/np.std
        (img)*std + mean_kido
```

```
9       return img
```

7.2　結果と考察

結果は図 6，7 のようになった．正規化を行うことで濃淡がはっきりとした．

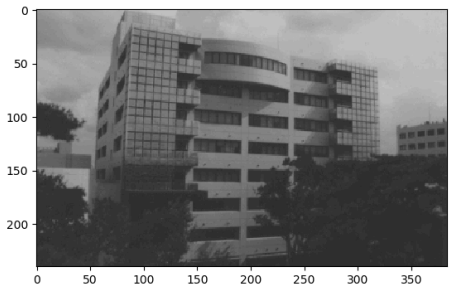

図 6　平均輝度 100, 輝度値の標準偏差 50 の画像.

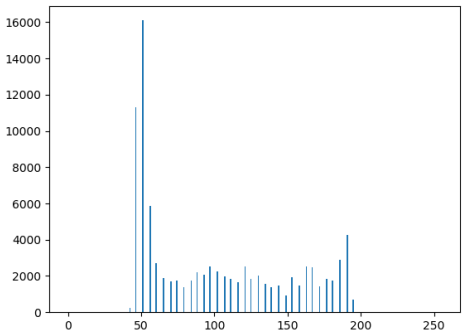

図 7　図 6 の画素値のヒストグラム.

8　課題 7 図形の面積の計算

図 8 中の 6 つの図形の面積，つまり画素数，を計算した.

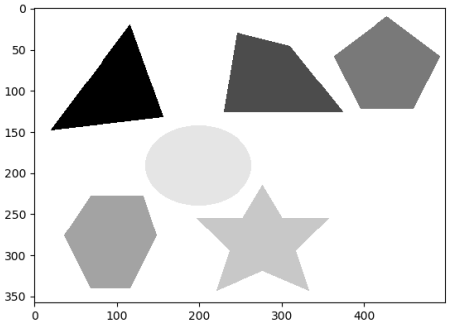

図 8　面積を求める画像.

8.1　手法

画素値集合を取って，どの画素値があるか把握した上で，その画素値の数をカウントして求めた.

ソースコード 8　kadai7.py

```
1  uniq = unique(gazo)
2  result = {
3      uniq[0]:0,
4      uniq[1]:0,
5      uniq[2]:0,
6      uniq[3]:0,
7      uniq[4]:0,
8      uniq[5]:0,
9      uniq[6]:0
10     }
11 for y in range(0,358):
12     for x in range(0,499):
13         value = gazo[y][x]
14         if value == uniq[0]:
15            result[uniq[0]] +=1
16         elif value == uniq[1]:
17            result[uniq[1]] +=1
18         elif value == uniq[2]:
19            result[uniq[2]] +=1
20         elif value == uniq[3]:
21            result[uniq[3]] +=1
22         elif value == uniq[4]:
23            result[uniq[4]] +=1
24         elif value == uniq[5]:
25            result[uniq[5]] +=1
26         elif value == uniq[6]:
27            result[uniq[6]] +=1
```

8.2　結果と考察

結果を表 1 に示した.

表 1　図 8 の各図形の面積.

図形	面積	画素値
三角形	8145	0
台形	9185	76
五角形	9265	200
六角形	9425	122
星	9545	164
丸	9841	229

8.3　課題 8 図形の輪郭線の長さの計算

画像 8 中の各図形の輪郭線の長さを計算した.

8.4　手法

ラプラシアンフィルタを用いて輪郭抽出を行う. しかしながら，ただ輪郭抽出を行うだけだと各図形の輪郭線の画素値が均一でないため課題 7 のように数えあげることができない. なので，なので次の手順で輪郭の長さを計算した.

1. 入力画像 A からラプラシアンフィルタを用いて輪郭線の画素値を 0，その他の部分を 255 にした画像 B を作成した.

2. $C = |B - A|$ を求めて，各図形の画素値が均一でないことを利用して，輪郭線画素値を不均一にした.

3. $D = |C - A|$ を求めて，輪郭内部の画素値を 0 にした. つまり，輪郭線以外の画素値は 0 である.

4. 課題 7 と同様に，各輪郭線の長さを求めた.

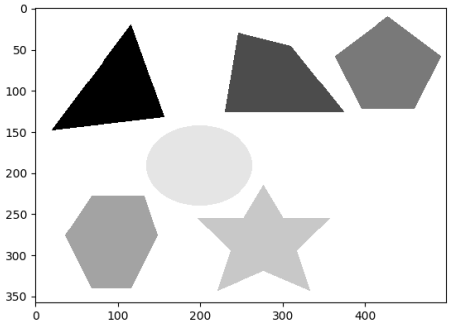

図 9　輪郭だけを持つ図形.

8.5　結果と考察

結果を表 2 に示した. ラプラシアンフィルタは周りの画素を用いて計算するので，当然条件によって輪郭線の画素値は変わってくるので，前処理を施した.

表 2　図 8 の各輪郭線の長さ.

図形	長さ
三角形	276
台形	320
五角形	320
六角形	336
星	384
丸	480

9　結論

画像処理の基礎実装を通して，画像処理について学ぶことができた. さらに，課題 8 では画像の前処理を考えることで，フィルタ演算だけではできない計算を行うことを学んだ.

10　感想

ディスプレイを濡れたティッシュで掃除すると 3 色に輝いてたのはとても面白かった.