画像処理実験 第 3 回

学生番号: 09B23523
大野颯也 (OHNO, Soya)

2025 年 10 月 18 日

1 第 3 回

1.1 課題 3.1

1.1.1 関数,b の作成

特徴座標 w が,

$$w = \begin{pmatrix} x_0 & y_0 & u_0 & v_0 \\ x_1 & y_1 & u_1 & v_1 \\ x_2 & y_2 & u_2 & v_2 \\ x_3 & y_3 & u_3 & v_3 \end{pmatrix}$$

であるので，行列 A,b は以下のようになる．

$$w = \begin{pmatrix} x_0 & y_0 & 1 & 0 & 0 & 0 & -u_0x_0 & -u_0y_0 \\ 0 & 0 & 0 & x_0 & y_0 & 1 & -u_0x_0 & -u_0y_0 \\ x_1 & y_1 & 1 & 0 & 0 & 0 & -u_1x_1 & -u_1y_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -v_1x_1 & -v_1y_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -u_2x_2 & -u_2y_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -v_2x_2 & -v_2y_2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -u_3x_3 & -u_3y_3 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -v_3x_3 & -v_3y_3 \end{pmatrix}$$

$$b^\top = \begin{pmatrix} u_0 & v_0 & u_1 & v_1 & u_2 & v_2 & u_3 & v_3 \end{pmatrix}$$

そして，以下がコードである．ここでは，後の課題で任意の個数の特徴点に合わせれるように，`N=w.shape[0]` で作成する行列の行数も変更するようにしてある．

```
 1: def makeAb(w):
 2:   N=w.shape[0]
 3:   A=np.zeros((N*2,8),'d')
 4:   b=np.zeros(N*2,'d')
 5:   i=np.arange(N)
 6:   A[i*2+0,0]=w[i,0]
 7:   A[i*2+0,1]=w[i,1]
 8:   A[i*2+0,2]=1
 9:   A[i*2+0,6]=-w[i,0]*w[i,2]
10:   A[i*2+0,7]=-w[i,1]*w[i,2]
11:   A[i*2+1,3]=w[i,0]
12:   A[i*2+1,4]=w[i,1]
13:   A[i*2+1,5]=1
14:   A[i*2+1,6]=-w[i,0]*w[i,3]
15:   A[i*2+1,7]=-w[i,1]*w[i,3]
```

```
16:    b[i*2    ]=w[i,2]
17:    b[i*2+1  ]=w[i,3]
18:    return A,b
```

実行した結果は以下のようになる．

```
1: [[256.00 218.00   1.00   0.00   0.00   0.00 -94976.00 -80878.00]
2:  [  0.00   0.00   0.00 256.00 218.00   1.00 -58880.00 -50140.00]
3:  [347.00 220.00   1.00   0.00   0.00   0.00 -160661.00 -101860.00]
4:  [  0.00   0.00   0.00 347.00 220.00   1.00 -79810.00 -50600.00]
5:  [263.00 367.00   1.00   0.00   0.00   0.00 -100729.00 -140561.00]
6:  [  0.00   0.00   0.00 263.00 367.00   1.00 -99677.00 -139093.00]
7:  [413.00 315.00   1.00   0.00   0.00   0.00 -218890.00 -166950.00]
8:  [  0.00   0.00   0.00 413.00 315.00   1.00 -135051.00 -103005.00]]
9: [371.00 230.00 463.00 230.00 383.00 379.00 530.00 327.00]
```

例と値が同じであるので，正しく動いている．

1.2 課題 3.2

1.2.1 calcHomography で計算した H を使った画像合成

図 1 が H を使って生成した合成画像である．

図 1: H を使った画像合成

1.2.2 合成画像の品質の改善

図 2 が特徴座標 w を変更して得られた合成画像である．

1.2.3 5 組以上の特徴点を使った合成画像

図 3a が特徴点の個数が 8 個の合成画像であり，図 3b が特徴点の個数が 12 個の合成画像である．以下に画像合成に使用した w を載せる．この w の内，上から 4 行が特徴点 4 個，8 行が特徴点 8 個，12 行が特徴

図 2: 品質が改善された合成画像

(a) 特徴点 8 個の合成画像

(b) 特徴点 12 個の合成画像

図 3: 特徴点数による合成画像の比較

点 12 個の場合である.

```
  w=np.array([
44,42,165,65,
627,17,746,15,
58,558,183,563,
625,538,754,554,
113,187,233,202,
462,208,577,215,
162,523,281,533,
458,471,577,483,
269,285,387,294,
387,292,502,300,
288,350,404,362,
396,369,513,379,
```

```
14: ]).reshape(-1,4)
```

特徴点は，4 個のとき (図 2) が最も離れた 4 つの角で，8 個のとき (図 3a) では真ん中と端の中間あたりで，12 個のとき (図 3b) では真ん中あたりに追加した．画像を比較すると，特徴点の個数が多いほど綺麗に合成されていることが分かる．

1.2.4 (u,v) の座標を自動修正する方法

```
 1: def sumOfSquaredDifference(w, im0, im1):
 2:   for i in range(w.shape[0]):
 3:     W=7
 4:     x,y,u,v=w[i]
 5:
 6:     tpl=im0[y-W:y+W+1,
 7:             x-W:x+W+1]
 8:     tgt=im1[v-W:v+W+1,
 9:             u-W:u+W+1]
10:     res=np.zeros((7,7))
11:     for dv in range(-3,4):
12:       for du in range(-3,4):
13:         tgt=im1[v+dv-W:v+dv+W+1,
14:                 u+du-W:u+du+W+1]
15:         res[dv+3,du+3]=((tpl-tgt)**2).sum()
16:     pos = np.array(np.unravel_index(np.argmin(res), res.shape))
17:     print(f"{i}[v,u] = {pos - [3,3]}")
```

上記のコードは，特徴点の周囲 7x7 ピクセルを比較して，最も差分が小さくなる (u,v) を探索するものである．特徴点の周囲 W ピクセルを切り出し，その中で (u,v) を中心に-3 から +3 までずらしたときの差分を計算し，最も差分が小さくなる (u,v) を選択している．この方法により，手動で指定した特徴点の (u,v) 座標を修正することができる．しかしこのコードでは差分を計算するのみで，実際に w の (u,v) を更新する部分は含まれていないため，一度実行したのちに手動で更新する必要がある．

実行前の w は以下の通りである．

```
 1:   w=np.array([
 2: 44,42,165,65,
 3: 627,17,746,15,
 4: 58,558,183,563,
 5: 625,538,754,554,
 6: 113,187,233,202,
 7: 462,208,577,215,
 8: 162,523,281,533,
 9: 458,471,577,483,
10: 269,285,387,294,
11: 387,292,502,300,
12: 288,350,404,362,
13: 396,369,513,379,
14: ]).reshape(-1,4)
```

実行した出力は以下の通りである．

```
1: 0[v,u] = [3 3]
2: 1[v,u] = [-3 -2]
3: 2[v,u] = [1 0]
4: 3[v,u] = [2 1]
5: 4[v,u] = [ 1 -1]
6: 5[v,u] = [1 1]
7: 6[v,u] = [-1  0]
8: 7[v,u] = [1 0]
```

```
 9: 8[v,u] = [ 3 -2]
10: 9[v,u] = [2 1]
11: 10[v,u] = [-1  0]
12: 11[v,u] = [1 0]
```

これを参考にして，w の (u,v) を修正すると以下のようになる

```
 1:   w=np.array([
 2: 44,42,165,65,
 3: 627,17,746,15,
 4: 58,558,183,564,
 5: 625,538,755,556,
 6: 113,187,232,203,
 7: 462,208,578,216,
 8: 162,523,281,532,
 9: 458,471,577,484,
10: 269,285,385,297,
11: 387,292,503,302,
12: 288,350,404,361,
13: 396,369,513,380,
14: ]).reshape(-1,4)
```

図 4a,4b が修正前と後の合成画像である．

(a) 自動修正前　　(b) 自動修正後

図 4: 自動修正前後の比較

これらの画像を比較すると，特に理科大の建物の文字の部分で修正前よりも綺麗に合成されていることが伺える．

1.2.5 撮影した画像の合成

使用する画像は図 5a,5b である．

使用する特徴点 w は以下の通りである ((u, v) の自動修正済み)．

```
 1:   w=np.array([
 2: 909,362,598,366,
 3: 854,738,556,743,
 4: 500,519,178,549,
 5: 377,303,19,303,
 6: ]).reshape(-1,4)
```

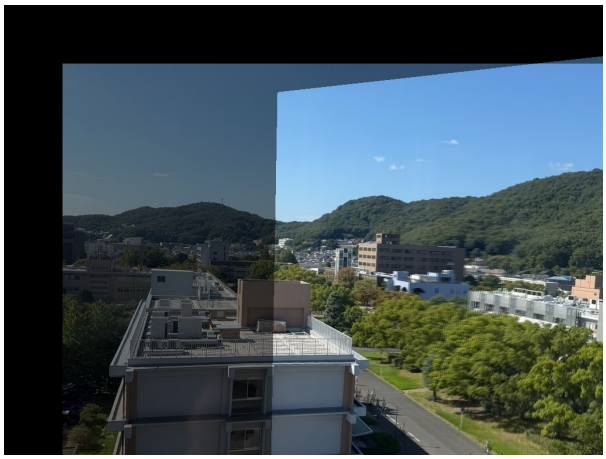
(a) 撮影画像 1

(b) 撮影画像 2

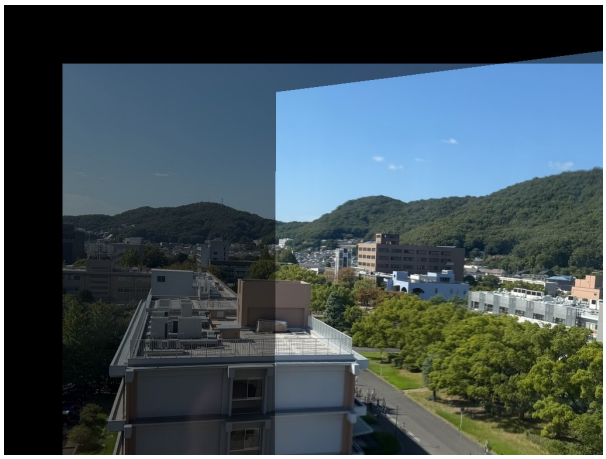
(c) 撮影画像 3

図 5: 3 枚の撮影画像の比較

そして，得られた合成画像が図 6a である．



(a) 撮影画像の合成結果



(b) 撮影画像の合成結果（特徴点 12 個）

図 6: 特徴点数による撮影画像の合成結果比較

特徴点の個数を 12 個に増やした合成画像が図 6b である．

1.2.6 重なりの少ない画像の合成

図 5a,5b,5c の 3 枚の画像を使用し，図 5a と図 5c を合成した．

使用した特徴点 w は以下の通りである．ただし，w1 が図 5a と図 5b の特徴点，w2 が図 5b と図 5c の特徴点である．

```
1:    w1=np.array([
2: 598,366,909,362,
3: 556,743,854,738,
4: 178,549,500,519,
5: 19,303,377,303,
6: 618,424,929,422,
7: 366,364,668,358,
8: 596,409,905,406,
9: 553,503,856,500,
10: 154,330,484,326,
11: 253,370,565,362,
12: 97,361,438,352,
13: 97,323,438,319,
14: ]).reshape(-1,4)
```

```
15:
16:    w2=np.array([
17:  83,699,455,668,
18:  83,608,458,585,
19:  85,513,463,499,
20:  19,368,415,368,
21:  569,360,953,372,
22:  541,290,923,295,
23:  376,299,734,306,
24:  166,368,535,371,
25:  580,563,960,601,
26:  598,638,979,687,
27:  589,239,983,237,
28:  273,234,635,243,
29:  ]).reshape(-1,4)
```

得られた合成画像が図 7a である．

また，3 枚間の画像の合成結果は図 7b のようである．

(a) 重なりの少ない画像の合成結果

(b) 3 枚間の画像の合成結果

図 7: 異なる重なり条件における画像合成結果の比較

2 感想