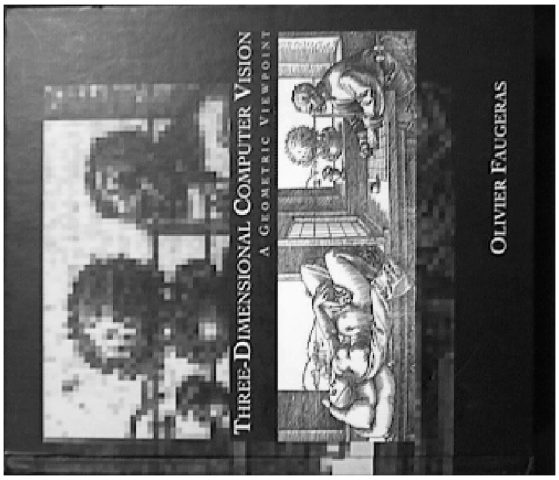
Computer Vision
CSCI-GA.2272-001
Assignment 3

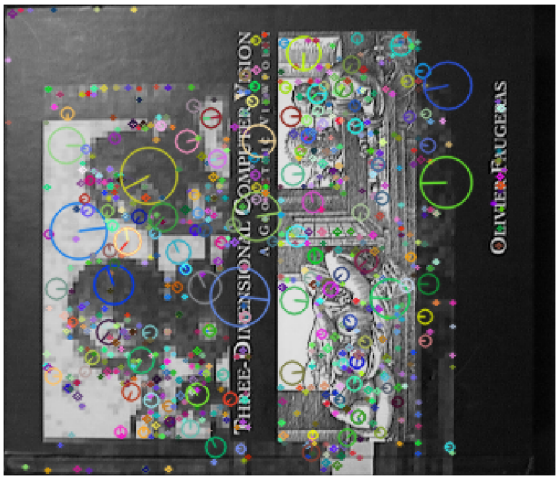
Chaojie Zhang
cz2064

1 Image Alignment

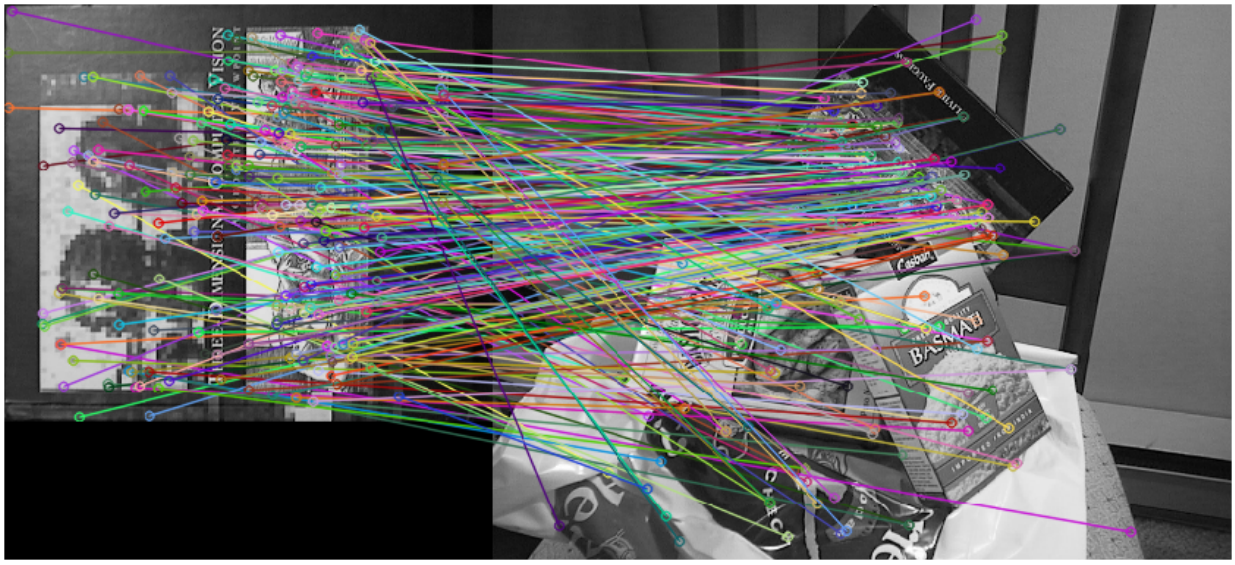
- Images(Book and Scene)



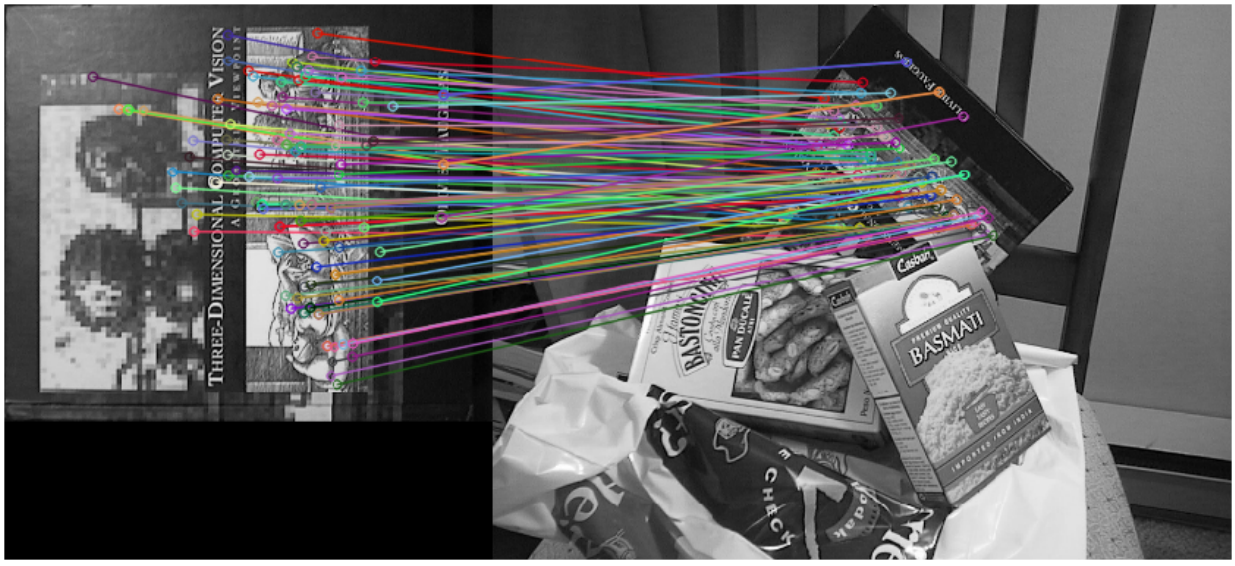
- SIFT Feature Detector



- Closest Neighbor



- RANSAC



- Homography Matrix H

The first H is the 3x3 homography matrix, the second H is reshaped from q, it is used in the cv2.warpAffine transform.

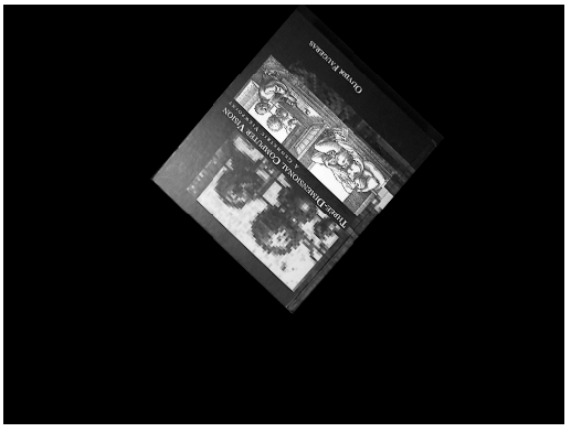
```
H = np.zeros([3,3])
H[0][0] = best_q[0]
H[0][1] = best_q[1]
H[0][2] = best_q[4]
H[1][0] = best_q[2]
H[1][1] = best_q[3]
H[1][2] = best_q[5]
H[2][2] = 1
H
```

```
array([[  0.40967485,   0.45021564,   0.42946945],
       [133.71993826,  -0.47768452, 158.69233577],
       [  0.        ,   0.        ,   1.        ]])
```

```
H = best_q.reshape(2,3)
H
```

```
array([[  0.40967485,   0.45021564, 133.71993826],
       [ -0.47768452,   0.42946945, 158.69233577]])
```

- Transform Image



2 Estimating the Camera Parameters

- Image Points and World Points

```
image_vector
```

```
array([[ 5.11770701,  5.5236545 ,  7.16310171,  5.22216628,  5.60479614,
        13.59494885,  8.73452189,  6.22433952,  9.74763886,  5.09031079],
       [ 4.76538441,  3.87032917,  7.35942066,  4.4279585 ,  4.67483648,
        10.05215495,  5.56420531,  3.90821885,  6.90423723,  4.5508513 ],
       [ 1.        ,  1.        ,  1.        ,  1.        ,  1.        ,
         1.        ,  1.        ,  1.        ,  1.        ,  1.        ]])
```

```
world_vector
```

```
array([[0.8518447 , 0.55793851, 0.81620571, 0.70368367, 0.71335444,
        0.1721997 , 0.04904683, 0.28614965, 0.13098247, 0.84767647],
       [0.75947939, 0.01423302, 0.97709235, 0.52206092, 0.2280389 ,
        0.96882014, 0.75533857, 0.25120055, 0.94081954, 0.20927164],
       [0.94975928, 0.59617708, 0.22190808, 0.93289706, 0.4496421 ,
        0.3557161 , 0.89481276, 0.93273619, 0.70185317, 0.45509169],
       [1.        , 1.        , 1.        , 1.        , 1.        ,
        1.        , 1.        , 1.        , 1.        , 1.        ]])
```

- Matrix A

```
A = np.zeros([2*image_vector.shape[1],12])
for i in range(image_vector.shape[1]):
    A[i][4:8] = -image_vector[2][i] * world_vector[:,i]
    A[i][8:12] = image_vector[1][i] * world_vector[:,i]
    A[10+i][:4] = image_vector[2][i]*world_vector[:,i]
    A[10+i][8:12] = -image_vector[0][i]*world_vector[:,i]
```

- Matrix P

```
U,D,V = np.linalg.svd(A)
```

```
P = V[-1:].reshape(3,4)
print(P)
```

```
[[ 1.27000127e-01  2.54000254e-01  3.81000381e-01  5.08000508e-01]
 [ 5.08000508e-01  3.81000381e-01  2.54000254e-01  1.27000127e-01]
 [ 1.27000127e-01 -2.77555756e-17  1.27000127e-01 -8.32667268e-17]]
```

- Reprojection Points and Image Points

```
np.round_(re_projection,3)
```

```
array([[ 5.118,  5.524,  7.163,  5.222,  5.605, 13.595,  8.735,  6.224,
         9.748,  5.09 ],
       [ 4.765,  3.87 ,  7.359,  4.428,  4.675, 10.052,  5.564,  3.908,
         6.904,  4.551],
       [ 1.   ,  1.   ,  1.   ,  1.   ,  1.   ,  1.   ,  1.   ,  1.   ,
         1.   ,  1.   ]])
```

```
np.round_(image_vector,3)
```

```
array([[ 5.118,  5.524,  7.163,  5.222,  5.605, 13.595,  8.735,  6.224,
         9.748,  5.09 ],
       [ 4.765,  3.87 ,  7.359,  4.428,  4.675, 10.052,  5.564,  3.908,
         6.904,  4.551],
       [ 1.   ,  1.   ,  1.   ,  1.   ,  1.   ,  1.   ,  1.   ,  1.   ,
         1.   ,  1.   ]])
```

- Compute C using SVD

```
U, s, V = np.linalg.svd(P)
C = V[-1]
C = C[:-1]/C[-1]
C
```

```
array([ 1., -1., -1.])
```

- Compute C using RQ decomposition

```
r,q = linalg.rq(P, mode='economic')
R = (q.T)[:-1].T
t = (q.T)[-1].T
C_ = np.linalg.solve(-R,t)
C_
```

```
array([ 1., -1., -1.])
```

3 Structure from Motion

- Image Points

```
matrix = sio.loadmat('./assignment3/sfm_points.mat')['image_points']
matrix.shape
```

```
(2, 600, 10)
```

- Translation Vector t

```
translation_vector = np.mean(matrix,axis=1)
translation_vector
```

```
array([[ 5.49560397e-17,  3.31216536e-17, -1.06118817e-16,
         4.27435864e-17, -9.25185854e-19, -7.75305746e-17,
         1.22124533e-17,  4.99600361e-18, -9.43689571e-18,
        -2.08166817e-18],
       [-7.03141249e-18,  3.97829917e-18,  1.24900090e-17,
        -1.85962357e-17, -3.99217696e-17,  8.83552490e-17,
         6.36527867e-17,  2.71773345e-18, -2.14643118e-17,
         3.85802501e-17]])
```

- Measurement Matrix W

```
matrix_centered = np.zeros(matrix.shape)
for i in range(matrix_centered.shape[0]):
    for j in range(matrix_centered.shape[2]):
        matrix_centered[i,:,j] = matrix[i,:,j] - translation_vector[i][j]
```

```
measurement_matrix = np.zeros([matrix.shape[0]*matrix.shape[2],matrix.shape[1]])
for i in range(matrix.shape[1]):
    for j in range(matrix.shape[2]):
            measurement_matrix[2*j,i] = matrix_centered[0,i,j]
            measurement_matrix[2*j+1,i] = matrix_centered[1,i,j]
```

- SVD Decomposition of W

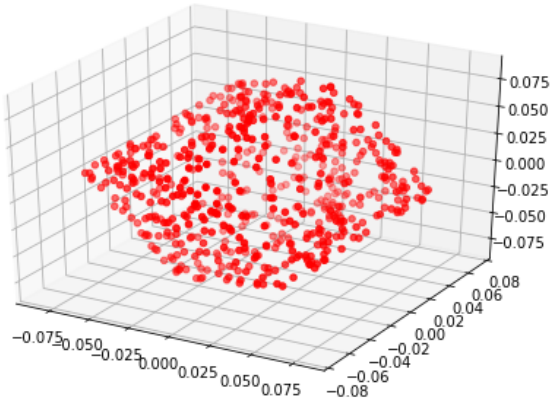
```
U,D,V = np.linalg.svd(measurement_matrix)
V = V.T
D = D * np.identity(D.shape[0])
```

- Camera Location M

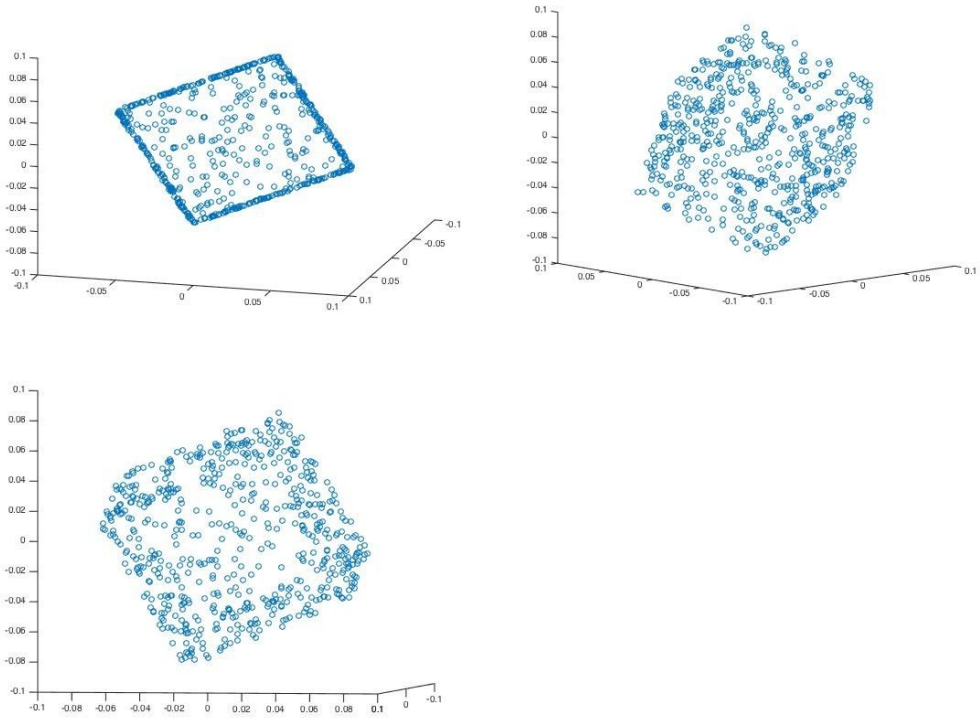
```
M = np.dot(U[:,:3],D[:3,:3])
M
```

```
array([[-7.50914219,  3.30837904, -3.71763726],
       [-4.53754376, -1.57773527,  7.74574759],
       [ 0.17858821, -8.56620251, -2.47587867],
       [ 9.05169424,  0.12603637,  0.70587237],
       [ 8.25306132,  2.16911022, -3.48212517],
       [-0.13132314, -7.68175234, -4.32518806],
       [-3.76826539, -8.34775199,  1.20087007],
       [ 8.27600638, -3.50666717,  0.57004455],
       [-0.73461089, -8.39784553, -2.88977146],
       [-8.50036578,  1.60529571, -2.55252038],
       [ 8.45690903, -2.56525708, -1.79392742],
       [-3.28948312, -6.10374195, -5.44642826],
       [-2.96665571, -7.78843781, -3.22986642],
       [ 8.45107965, -1.64131526, -2.78078037],
       [-1.4368307 , -8.62307292,  3.07678742],
       [-7.95142326, -0.23710514, -4.1742912 ],
       [ 8.6277954 , -2.12325785, -1.6361374 ],
       [-0.41749971,  4.10544054, -8.14813897],
       [ 7.44257036, -3.77728996,  3.4002285 ],
       [-5.22854825, -5.82482627,  5.11580038]])
```

- 3D World Points Locations



- 3D World Points Locations (Matlab)



M and t in The First Camera

```
print(M[:2])
```

```
[[-7.50914219  3.30837904 -3.71763726]
 [-4.53754376 -1.57773527  7.74574759]]
```

```
print(translation_vector[:,0])
```

```
[ 5.49560397e-17 -7.03141249e-18]
```

First 10 World Points

```
print(_3D_points[:10])
```

```
[[ 0.00577163  0.06460628 -0.02497615]
 [ 0.0005761   0.06885363 -0.03458151]
 [-0.04293585  0.06330479  0.02861711]
 [ 0.04745038  0.04904207 -0.01257547]
 [-0.04210186  0.06789239  0.01175164]
 [ 0.05961964  0.0460518  -0.01438374]
 [ 0.00909167  0.06002049 -0.01229997]
 [ 0.01039489  0.04602065  0.03529275]
 [-0.02589081  0.05702972  0.03337375]
 [ 0.01745598  0.04054264  0.04731859]]
```