ECE 661 Homework 1

Weichen Xu

xu1363@purdue.edu

1. Basic Methods

Mapping of homogeneous coordinates

If we already know the homography matrix H from one image to another image, we can obtain following function of mapping a point's homogeneous coordinates to its corresponding homogeneous coordinates in another image.

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = H \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

Homography Matrix

A key part of mapping is homography matrix, we usually obtain the homography matrix by using point pairs in both images.

Since we know that a typical representation of homography matrix is a 3*3 matrix with the h_{33} to be 1.

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix}$$

Applying the homography matrix to the mapping formula, we can get following result

$$x' = h_{11}x + h_{12}y + h_{13}z$$
$$y' = h_{21}x + h_{22}y + h_{23}z$$
$$z' = h_{31}x + h_{32}y + z$$

If we set $z = 1$ and $z' = 1$, then we can get following equations

$$x' = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + 1}$$
$$y' = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + 1}$$

Further we can further transform to following equations

$$x' = h_{11}x + h_{12}y + h_{13} - h_{31}xx' - h_{32}yx'$$
$$y' = h_{21}x + h_{22}y + h_{23} - h_{31}xy' - h_{32}yy'$$

In total, the H matrix has 8 unknown elements, so we need at least 4 sets of corresponding point pairs (x_i, y_i) and (x_i', y_i') with 8 equations to solve the equations.

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x_1' & -y_1x_1' \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y_1' & -y_1y_1' \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2x_2' & -y_2x_2' \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2y_2' & -y_2y_2' \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x_3x_3' & -y_3x_3' \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -x_3y_3' & -y_3y_3' \\ x_4 & y_4 & 1 & 0 & 0 & 0 & -x_4x_4' & -y_4x_4' \\ 0 & 0 & 0 & x_4 & y_4 & 1 & -x_4y_4' & -y_4y_4' \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} = \begin{bmatrix} x_1' \\ y_1' \\ x_2' \\ y_2' \\ x_3' \\ y_3' \\ x_4' \\ y_4' \end{bmatrix}$$

$$Ah = b$$

We can then obtain h by multiplying the inverse matrix of A with b

$$h = A^{-1}b$$

2. Implementation Process

Task 1.1

Step 1: Find the four corner points in three painting images and record their coordinates. Then choose a region on cat image with four corner points and record their coordinates.

Step 2: From the corresponding four point pairs of the source image and target image, we can calculate the homography matrix H

Step 3: Calculate the mapping from the points in desired area of painting image, to the point in cat image, and replace the pixel value of points in painting image with the corresponding nearest point' pixel value.

Task 1.2

Step 1: Respectively calculate the homography matrix of painting 1 mapping to painting 2 and painting 2 mapping to painting 3, denoted as $H_1$and H_2

Step 2: Get the homography matrix mapping painting 1 to painting 3 by $H = H_2H_1$

Step 3: Map the points in painting 3 with the homography matrix H

3. Original Images and Results

Task 1.1

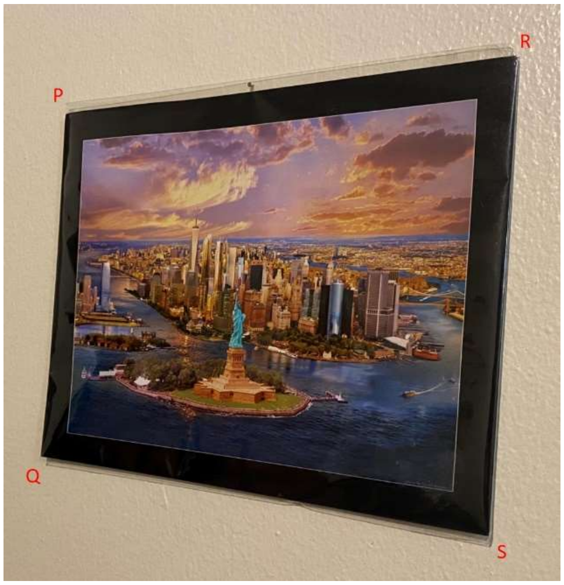

Figure 1. Original painting 1 with PQSR points

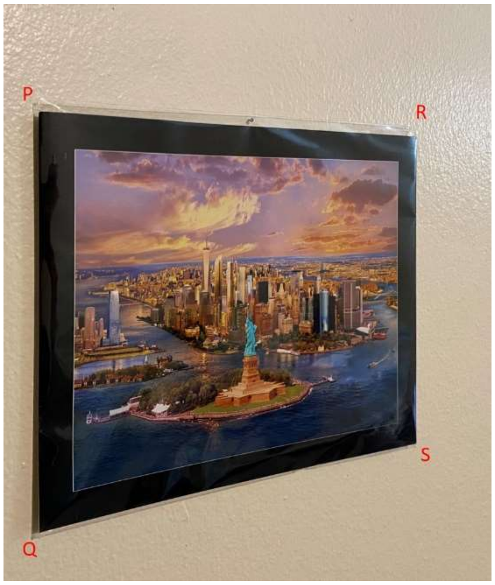

Figure 2. Original painting 2 with PQSR points

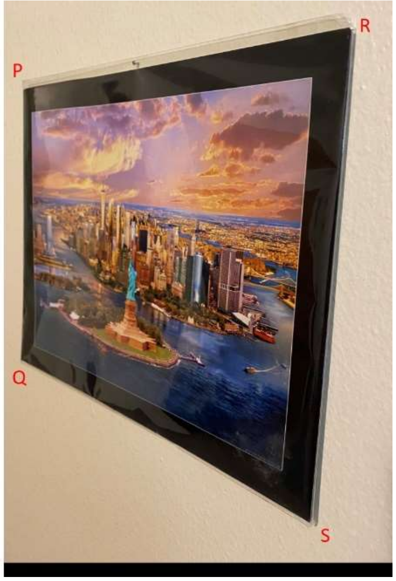

Figure 3. Original painting 3 with PQSR points

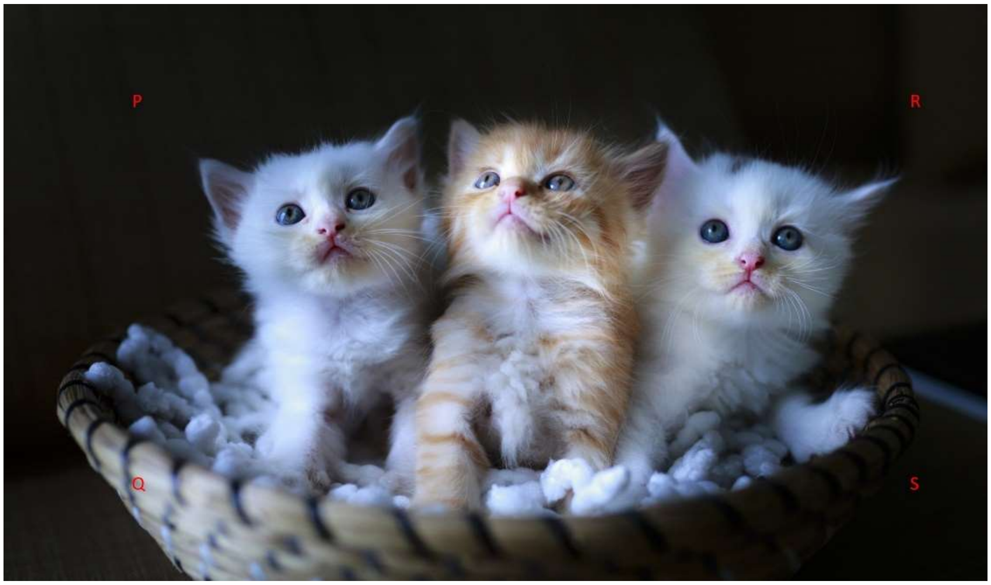

Figure 4. Cat picture with PQSR points

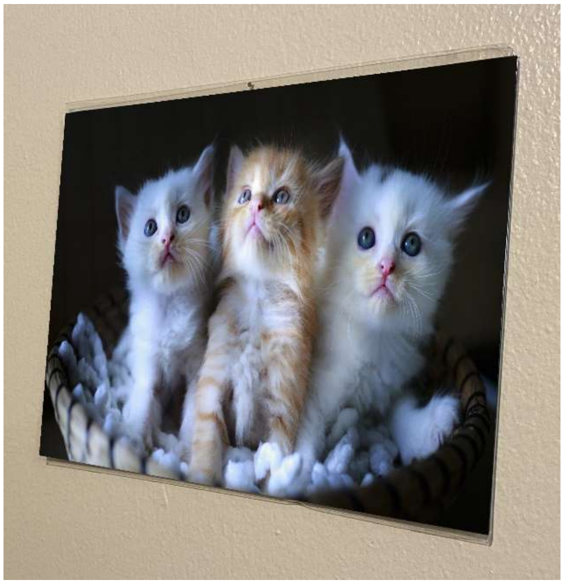

Figure 5. Mapping cat image to painting 1

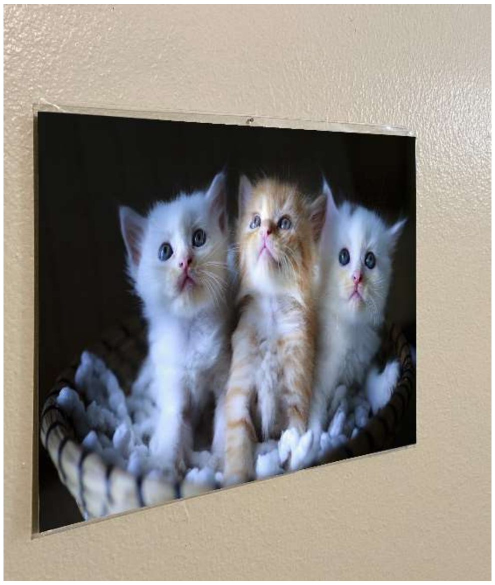

Figure 6. Mapping cat image to painting 2

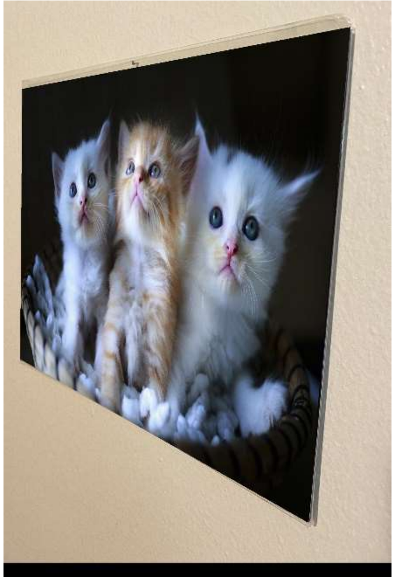

Figure 7. Mapping cat image to painting 3

Task 1.2

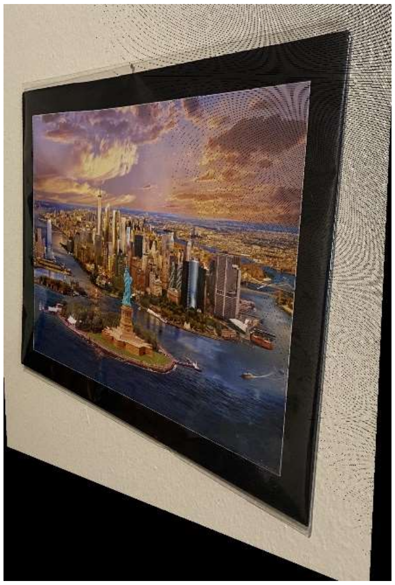

Figure 8. Mapping painting 1 to painting 3

Task 2.1

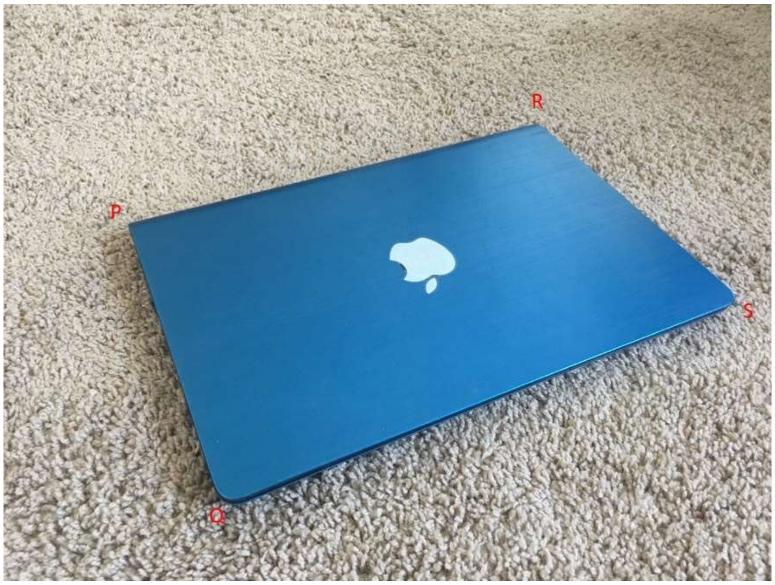

Figure 9. My image of laptop plane 1

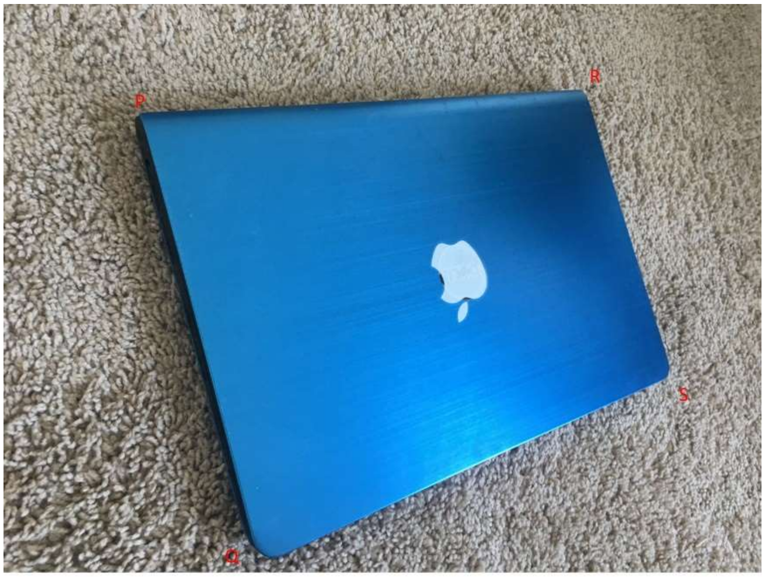

Figure 10. My image of laptop plane 2

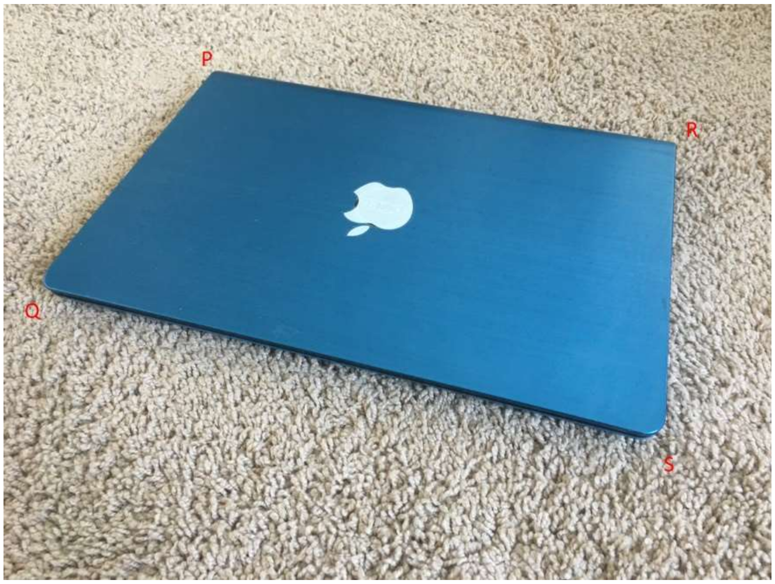

Figure 11. My image of laptop plane 3

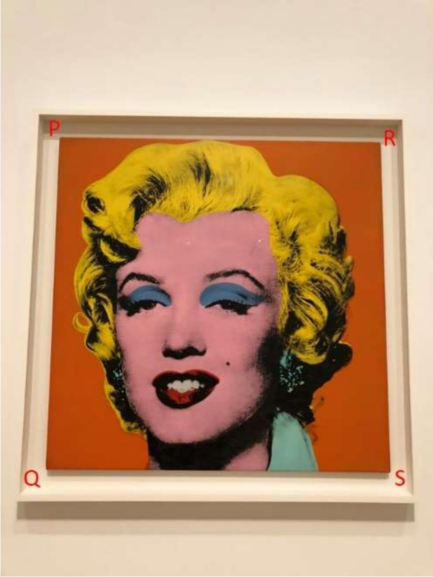

Figure 12. Face image

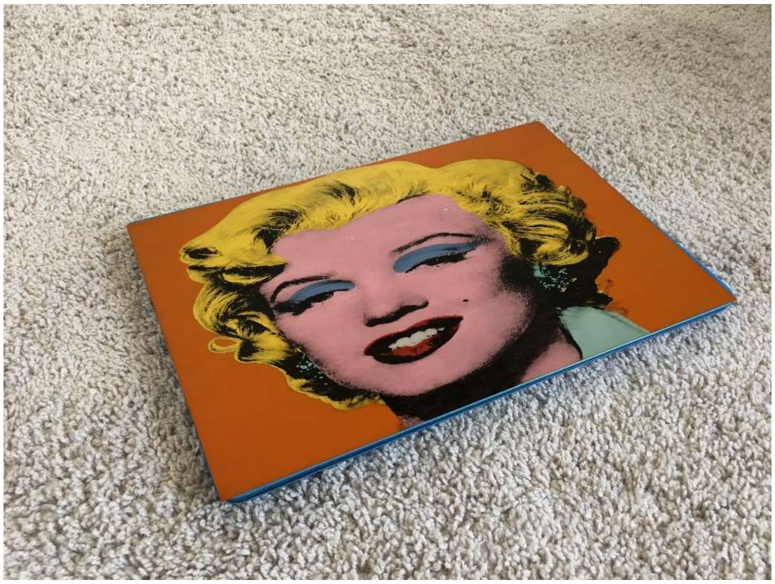

Figure 13. Mapping the face to laptop plane 1

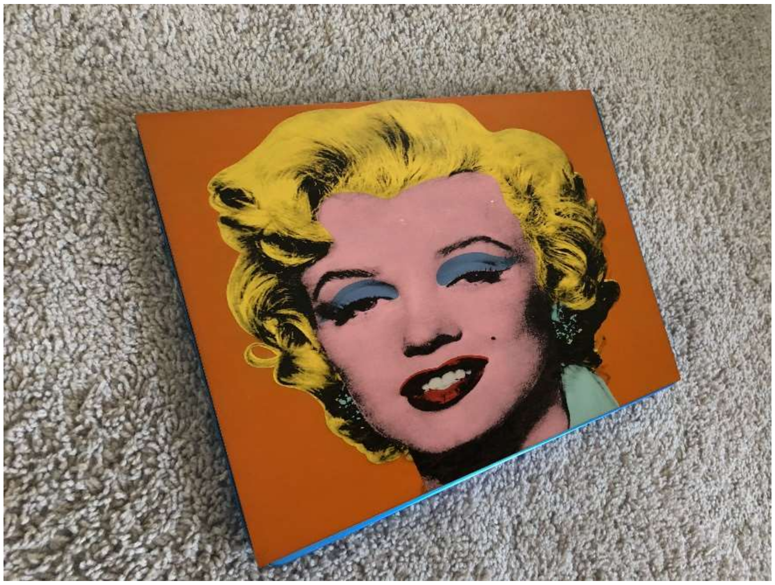

Figure 14. Mapping the face to laptop plane 2

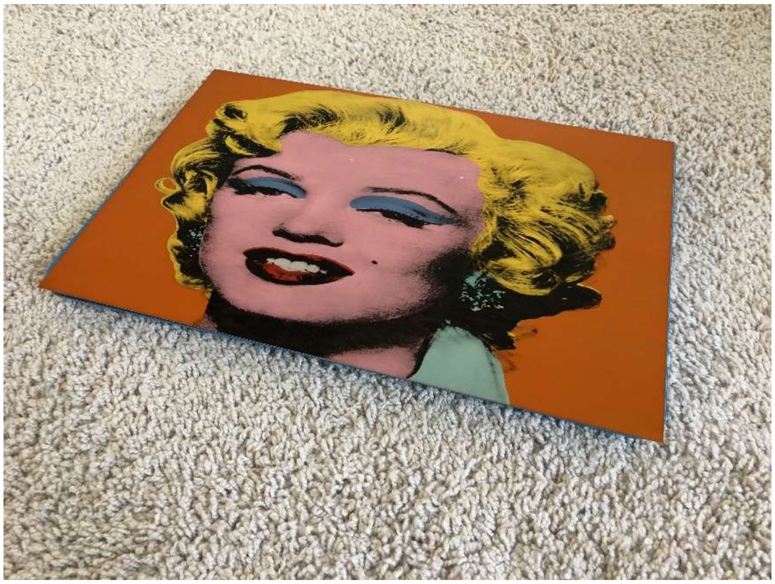

Figure 15. Mapping the face to laptop plane 3

Task 2.2

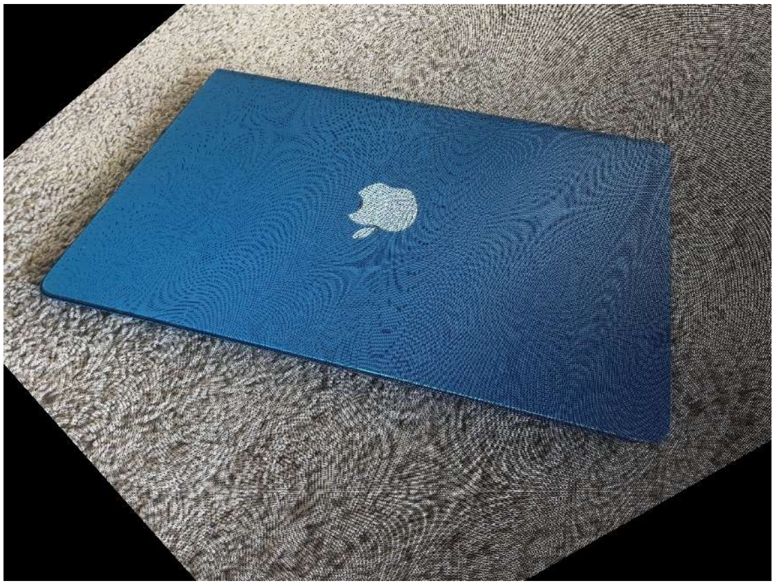

Figure 16. Mapping laptop plane 1 to laptop plane 3

4. Source Code

```
#!/usr/bin/env python
# coding: utf-8

# In[48]:


import cv2
import matplotlib.pyplot as plt
import numpy as np


# In[49]:


def get_homography(target,source):

    A = np.zeros((8,8))
    b = np.reshape(target,(8,1))
    for i in range(4):

        # define corresponding points pair of PQSR
        x1 = source[i,0]
        y1 = source[i,1]
        x2 = target[i,0]
        y2 = target[i,1]

        # build the row of matrix A
        A[i*2,:] = [x1,y1,1,0,0,0,-x1*x2,-y1*x2]
        A[i*2+1,:] = [0,0,0,x1,y1,1,-x1*y2,-y1*y2]

        # build the element of vector b
        b[i*2] = x2
        b[i*2+1] = y2

    # multiply the inverse matrix of A with b to get the vector of all coefficients of H
    h = np.matmul(np.linalg.inv(A),b)

    # attach the last scalar of H, which is 1
    h = np.append(h,1)

    # resize to 3*3 matrix
    H = np.reshape(h,(3,3))
    return H


# In[50]:


def mapping(img_target,img_source,target,source,H):

    # build a matrix with the same size of target image,
    # and fill all valid area within the four points PQSR with value 255
    area = np.zeros(img_target.shape[:],dtype = np.uint8)
    border = np.expand_dims(source,axis=0)
    cv2.fillPoly(area,border,255)

    img_new = img_target
    for j in range(img_target.shape[0]):
        for i in range(img_target.shape[1]):
            if area[j][i][0] == 255:
                input = np.array([i,j,1]).reshape((3,1))

                # get the corresponding location after applying homography
                output = np.matmul(H,input)
                x2 = np.int(np.round(output[0]/output[2]))
                y2 = np.int(np.round(output[1]/output[2]))

                if (x2>0) and (x2<img_source.shape[1]) and (y2>0) and (y2<img_source.shape[0]):
                    # replace the pixel value with corresponding points from source image
                    img_new[j,i,:] = img_source[y2,x2,:]
    return img_new


# In[51]:

# define the four point PQSR coordinates in each image of Task 1
img1 = np.array([[232,411],[134,1694],[1817,1991],[1926,195]])
img2 = np.array([[164,515],[174,2519],[1971,2095],[1968,635]])
img3 = np.array([[74,350],[66,1420],[1204,2067],[1372,105]])
img_cat = np.array([[100,100],[100,1000],[1800,1000],[1800,100]])


# In[52]:

# Task 1-a

# Define image location and read it with OpenCV
directory = "C:/Users/clare/OneDrive - purdue.edu/Desktop/ECE 661/hw2_Task1_Images/"
file_target = "painting1.jpeg"
file_source = "kittens.jpeg"
img_target = cv2.imread(directory+file_target,cv2.IMREAD_COLOR)
img_source = cv2.imread(directory+file_source,cv2.IMREAD_COLOR)

# get the homography matrix
H = get_homography(img_cat,img1)

# map the source image to the target image with homography matrix
img_new = mapping(img_target,img_source,img_cat,img1,H)

plt.imshow(cv2.cvtColor(img_new, cv2.COLOR_BGR2RGB))
plt.show()
cv2.imwrite(directory+"task1a.jpeg", img_new) # save image


# In[53]:

# Task 1-b
directory = "C:/Users/clare/OneDrive - purdue.edu/Desktop/ECE 661/hw2_Task1_Images/"
```

```
file_target = "painting2.jpeg"
file_source = "kittens.jpeg"
img_target = cv2.imread(directory+file_target,cv2.IMREAD_COLOR)
img_source = cv2.imread(directory+file_source,cv2.IMREAD_COLOR)

H = get_homography(img_cat,img2)
img_new = mapping(img_target,img_source,img_cat,img2,H)
plt.imshow(cv2.cvtColor(img_new, cv2.COLOR_BGR2RGB))
plt.show()
cv2.imwrite(directory+"task1b.jpeg", img_new)


# In[54]:

# Task 1-c
directory = "C:/Users/clare/OneDrive - purdue.edu/Desktop/ECE 661/hw2_Task1_Images/"
file_target = "painting3.jpeg"
file_source = "kittens.jpeg"
img_target = cv2.imread(directory+file_target,cv2.IMREAD_COLOR)
img_source = cv2.imread(directory+file_source,cv2.IMREAD_COLOR)

H = get_homography(img_cat,img3)
img_new = mapping(img_target,img_source,img_cat,img3,H)
plt.imshow(cv2.cvtColor(img_new, cv2.COLOR_BGR2RGB))
plt.show()
cv2.imwrite(directory+"task1c.jpeg", img_new)


# In[55]:

# Task 1-2
directory = "C:/Users/clare/OneDrive - purdue.edu/Desktop/ECE 661/hw2_Task1_Images/"
file_target = "painting1.jpeg"
img_target = cv2.imread(directory+file_target,cv2.IMREAD_COLOR)

# get the homography from image 1 to image 3 by multyplying H2 and H1
H1 = get_homography(img2,img1)
H2 = get_homography(img3,img2)
H = np.matmul(H2,H1)
img_new = np.zeros(img_target.shape,dtype = np.uint8)

for j in range(img_target.shape[0]):
    for i in range(img_target.shape[1]):
      input = np.array([i,j,1]).reshape((3,1))
      output = np.matmul(H,input)
      x2 = np.int(np.round(output[0]/output[2]))
      y2 = np.int(np.round(output[1]/output[2]))

      if (x2>0) and (x2<img_target.shape[1]) and (y2>0) and (y2<img_target.shape[0]):
        img_new[y2,x2,:] = img_target[j,i,:]

plt.imshow(cv2.cvtColor(img_new, cv2.COLOR_BGR2RGB))
plt.show()
cv2.imwrite(directory+"task1-2.jpeg", img_new)
```

```
# In[56]:

# define the four point PQSR coordinates in each image of Task 2
img1 = np.array([[641,1161],[1143,2615],[3853,1551],[2809,612]])
img2 = np.array([[683,574],[1339,2958],[3556,1969],[3080,447]])
img3 = np.array([[1107,340],[176,1486],[3463,2308],[3524,725]])
img_face = np.array([[414,959],[319,3284],[2722,3302],[2654,996]])


# In[57]:

# Task 2-a
directory = "C:/Users/clare/OneDrive - purdue.edu/Desktop/ECE 661/hw2_Task1_Images/"
file_target = "mypic1.JPG"
file_source = "face.JPG"
img_target = cv2.imread(directory+file_target,cv2.IMREAD_COLOR)
img_source = cv2.imread(directory+file_source,cv2.IMREAD_COLOR)

H = get_homography(img_face,img1)
img_new = mapping(img_target,img_source,img_face,img1,H)
plt.imshow(cv2.cvtColor(img_new, cv2.COLOR_BGR2RGB))
plt.show()
cv2.imwrite(directory+"task2a.jpeg", img_new)


# In[59]:

# Task 2-b
directory = "C:/Users/clare/OneDrive - purdue.edu/Desktop/ECE 661/hw2_Task1_Images/"
file_target = "mypic2.JPG"
file_source = "face.JPG"
img_target = cv2.imread(directory+file_target,cv2.IMREAD_COLOR)
img_source = cv2.imread(directory+file_source,cv2.IMREAD_COLOR)

H = get_homography(img_face,img2)

img_new = mapping(img_target,img_source,img_face,img2,H)
plt.imshow(cv2.cvtColor(img_new, cv2.COLOR_BGR2RGB))
plt.show()
cv2.imwrite(directory+"task2b.jpeg", img_new)


# In[60]:

# Task 2-c
directory = "C:/Users/clare/OneDrive - purdue.edu/Desktop/ECE 661/hw2_Task1_Images/"
file_target = "mypic3.JPG"
file_source = "face.JPG"
img_target = cv2.imread(directory+file_target,cv2.IMREAD_COLOR)
img_source = cv2.imread(directory+file_source,cv2.IMREAD_COLOR)

H = get_homography(img_face,img3)
img_new = mapping(img_target,img_source,img_face,img3,H)
plt.imshow(cv2.cvtColor(img_new, cv2.COLOR_BGR2RGB))
plt.show()
cv2.imwrite(directory+"task2c.jpeg", img_new)


# In[61]:

# Task 2-2
directory = "C:/Users/clare/OneDrive - purdue.edu/Desktop/ECE 661/hw2_Task1_Images/"
file_target = "mypic1.JPG"
img_target = cv2.imread(directory+file_target,cv2.IMREAD_COLOR)

H1 = get_homography(img2,img1)
H2 = get_homography(img3,img2)
H = np.matmul(H2,H1)
img_new = np.zeros(img_target.shape,dtype = np.uint8)

for j in range(img_target.shape[0]):
    for i in range(img_target.shape[1]):
        input = np.array([i,j,1]).reshape((3,1))
        output = np.matmul(H,input)
        x2 = np.int(np.round(output[0]/output[2]))
        y2 = np.int(np.round(output[1]/output[2]))

        if (x2>0) and (x2<img_target.shape[1]) and (y2>0) and (y2<img_target.shape[0]):
            img_new[y2,x2,:] = img_target[j,i,:]

plt.imshow(cv2.cvtColor(img_new, cv2.COLOR_BGR2RGB))
plt.show()
cv2.imwrite(directory+"task2-2.jpeg", img_new)
```