**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.

**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 to be 1.

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

If we set and , then we can get following equations

Further we can further transform to following equations

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

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

Step 2: Get the homography matrix mapping painting 1 to painting 3 by

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

**3. Original Images and Results**

**Task 1.1**

A picture containing gallery, room, scene, television

Description automatically generated

Figure 1. Original painting 1 with PQSR points

A flat screen television

Description automatically generated

Figure 2. Original painting 2 with PQSR points

A picture containing oven, photo, different, microwave

Description automatically generated

Figure 3. Original painting 3 with PQSR points

A cat sitting on top of each other

Description automatically generated

Figure 4. Cat picture with PQSR points

A cat in a box

Description automatically generated

Figure 5. Mapping cat image to painting 1

A cat sitting in front of a mirror posing for the camera

Description automatically generated

Figure 6. Mapping cat image to painting 2

A cat in a box

Description automatically generated

Figure 7. Mapping cat image to painting 3

**Task 1.2**

A picture containing indoor, photo, box, oven

Description automatically generated

Figure 8. Mapping painting 1 to painting 3

**Task 2.1**

A picture containing computer

Description automatically generated

Figure 9. My image of laptop plane 1

A close up of a piece of paper

Description automatically generated

Figure 10. My image of laptop plane 2

A picture containing text, computer

Description automatically generated

Figure 11. My image of laptop plane 3

A picture containing room

Description automatically generated

Figure 12. Face image

A close up of a card

Description automatically generated

Figure 13. Mapping the face to laptop plane 1

A close up of a piece of paper

Description automatically generated

Figure 14. Mapping the face to laptop plane 2



Figure 15. Mapping the face to laptop plane 3

**Task 2.2**

A picture containing rug

Description automatically generated

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)