Computer Vision HW3 Report

B09901142 電機三 呂睿超

1. Part 1 Homography estimation

   Paste the warped canvas

2. Part 2 Marker-Based Planar AR

   (1) Paste the function code solve_homography(u, v) & warping( ) (both forward & backward)

   a. solve_homography(u,v)

```python
def solve_homography(u, v):
    """
    This function should return a 3-by-3 homography matrix,
    u, v are N-by-2 matrices, representing N corresponding points for v = T(u)
    :param u: N-by-2 source pixel location matrices
    :param v: N-by-2 destination pixel location matrices
    :return:
    """
    N = u.shape[0]
    H = None

    if v.shape[0] is not N:
        print('u and v should have the same size')
        return None
    if N < 4:
        print('At least 4 points should be given')

    # TODO: 1.forming A
    A = []
    for i in range(N):
        A.append([u[i][0],u[i][1],1,0,0,0,-u[i][0]*v[i][0],-u[i][1]*v[i][0],-v[i][0]])
        A.append([0,0,0,u[i][0],u[i][1],1,-u[i][0]*v[i][1],-u[i][1]*v[i][1],-v[i][1]])
    A = np.array(A)
    # TODO: 2.solve H with A
    _,_,V_t = np.linalg.svd(A) # do a to svd, A = US(V_t)
    #print(V_t)
    H = V_t[-1,:].reshape(3,3)
    #print(H)
    return H
```

b. warping

```python
def warping(src, dst, H, ymin, ymax, xmin, xmax, direction='b'):
    h_src, w_src, ch = src.shape
    h_dst, w_dst, ch = dst.shape
    H_inv = np.linalg.inv(H)

    # TODO: 1.meshgrid the (x,y) coordinate pairs
    # TODO: 2.reshape the destination pixels as N x 3 homogeneous coordinate
    xc, yc = np.meshgrid(np.arange(xmin, xmax, 1), np.arange(ymin, ymax, 1), sparse = False)
    xrow = xc.reshape(( 1,(xmax-xmin)*(ymax-ymin) ))
    yrow = yc.reshape(( 1,(xmax-xmin)*(ymax-ymin) ))
    onerow =  np.ones(( 1,(xmax-xmin)*(ymax-ymin) ))
    v = np.concatenate((xrow, yrow, onerow), axis = 0)

    if direction == 'b':
        # TODO: 3.apply H_inv to the destination pixels and retrieve (u,v) pixels, then reshape to (ymax-ymin),(xmax-xmin)
        v = np.dot(H_inv,v)
        #v = np.divide(v, v[-1,:])
        v[:2,:] = v[:2,:] / v[2,:][np.newaxis,:]
        srcy = np.round( v[1,:].reshape((ymax-ymin, xmax-xmin)) ).astype(int)
        srcx = np.round( v[0,:].reshape((ymax-ymin, xmax-xmin)) ).astype(int)

        # TODO: 4.calculate the mask of the transformed coordinate (should not exceed the boundaries of source image)
        # TODO: 5.sample the source image with the masked and reshaped transformed coordinates
        h_mask1 = (0<srcy)*(srcy<h_src)
        w_mask = np.where((srcx < 0) | (srcx >= w_src) )[:]
        mask1 = np.ones(srcx.shape, dtype=bool)
        mask1[w_mask] = False

        h_mask = np.where((srcy < 0) | (srcy >= h_src) )[:]
        mask2 = np.ones(srcy.shape, dtype=bool)
        mask2[h_mask] = False
        #print(h_mask1[0][0],mask2[0][0])
        #print(mask1)
        #w_mask = (0<srcx)*(srcx<w_src)
        mask   = mask1*mask2

        # TODO: 6. assign to destination image with proper masking
        dst[yc[mask], xc[mask]] = src[srcy[mask], srcx[mask]]
```

```python
    elif direction == 'f':
        # TODO: 3.apply H to the source pixels and retrieve (u,v) pixels, then reshape to (ymax-ymin),(xmax-xmin)
        v = np.dot(H,v)
        v[:2,:] = v[:2,:] / v[2,:][np.newaxis,:]
        dsty = np.round(v[1,:].reshape(ymax-ymin,xmax-xmin)).astype(int)
        dstx = np.round(v[0,:].reshape(ymax-ymin,xmax-xmin)).astype(int)

        # TODO: 4.calculate the mask of the transformed coordinate (should not exceed the boundaries of destination image)
        # TODO: 5.filter the valid coordinates using previous obtained mask
        # TODO: 6. assign to destination image using advanced array indicing
        dst[np.clip(dsty, 0, dst.shape[0]-1), np.clip(dstx, 0, dst.shape[1]-1)] = src

    return dst
```

(2) Briefly introduce the interpolation method you use

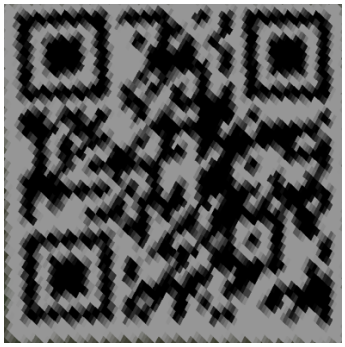
I used nearest_neighbor interpolation by finding the closest integer(np.round().astype(int))

3. Part 3 Unwarp the secret

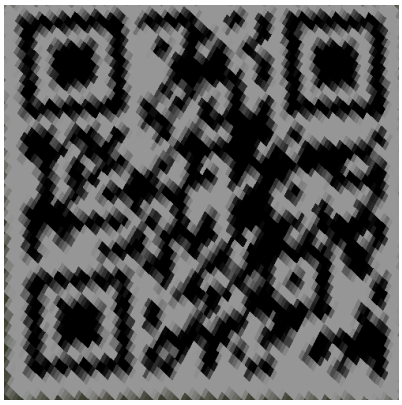
The secret link is http://media.ee.ntu.edu.tw/courses/cv/21S/

(1) Paste the 2 warped images

a. BL_secret1



b. BL_secret2



(2) Discuss the difference between 2 source images, are the warped results the same or different?

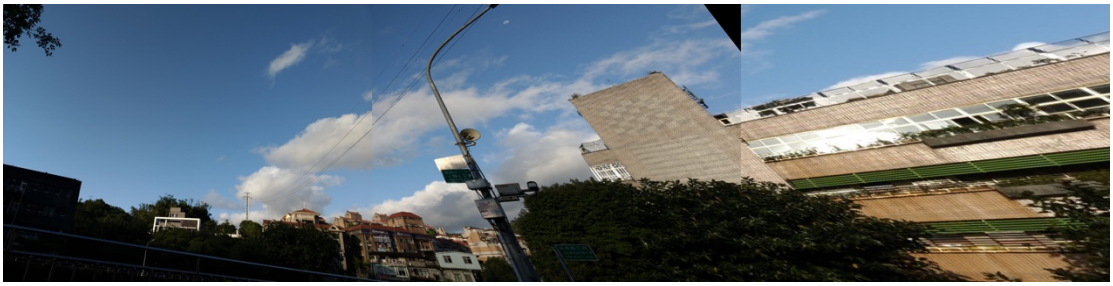
Yes, they are slightly different. The output of BL_secret2 is a little more vague.

(3) If the results are different, explain why.

The original image of BL_secret2 is a little curved and it will cause the warping to lose information because some of the pixels would be deleted by the mask operation

4. Part4 Panorama

(1) Paste your stitched panorama



(2) Can all consecutive images be stitched into a panorama

No

(3) If yes, explain your reason. If not, explain under what conditions will result in a failure?

If the images have no overlapping regions to provide matching points, it will cause failure.