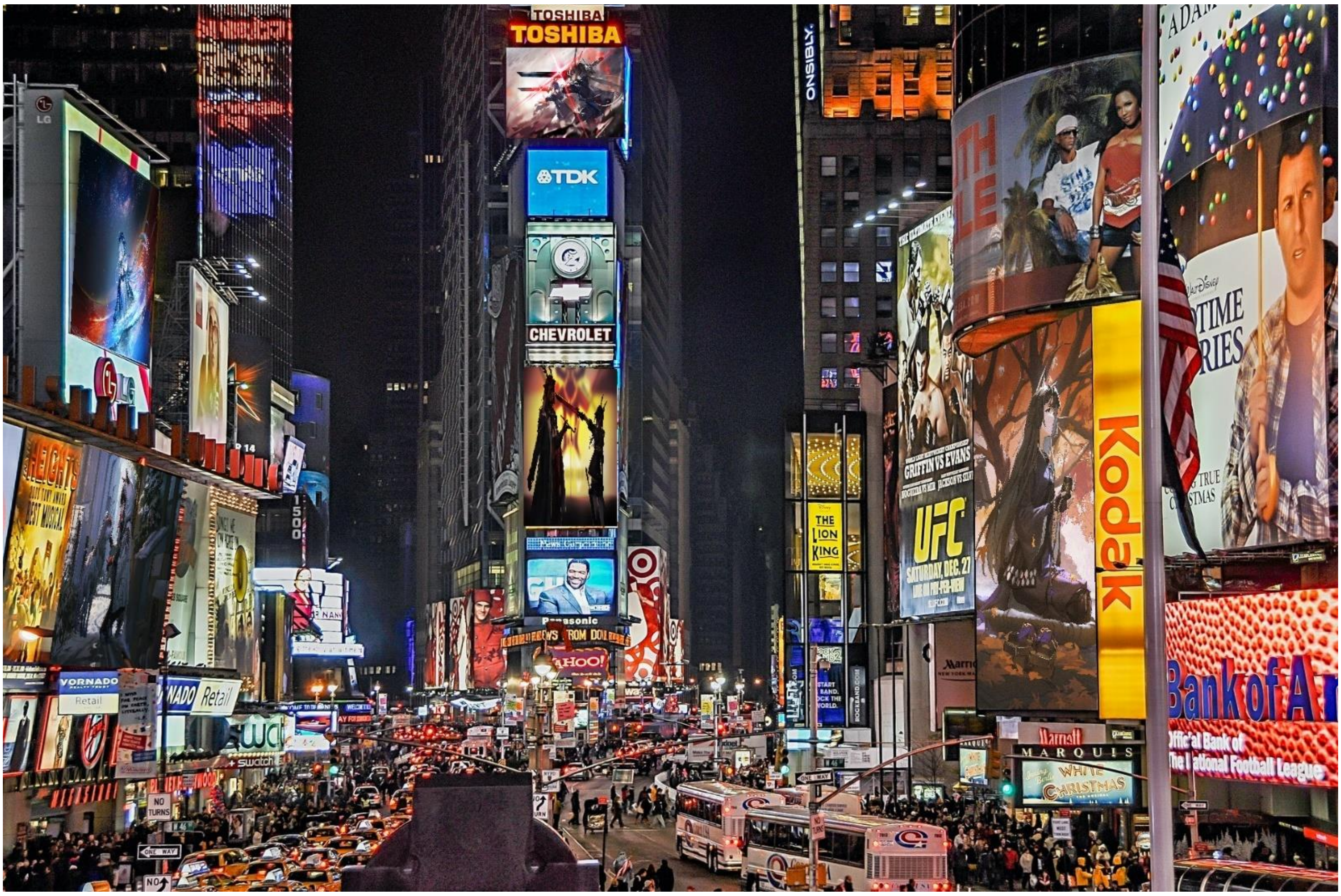
Computer Vision HW3 Report

Student ID: B10901074
Name: 曾柏穎

Part 1.

- **Paste your warped canvas**



Part 2.

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

```
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 = np.zeros((2*N, 9))
    for i in range(N):
        A[2*i] = np.array([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[2*i+1] = np.array([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]])
    # TODO: 2.solve H with A
        U, sigma, V = np.linalg.svd(A)
        H = np.reshape(V[-1], (3, 3))
    return H
```

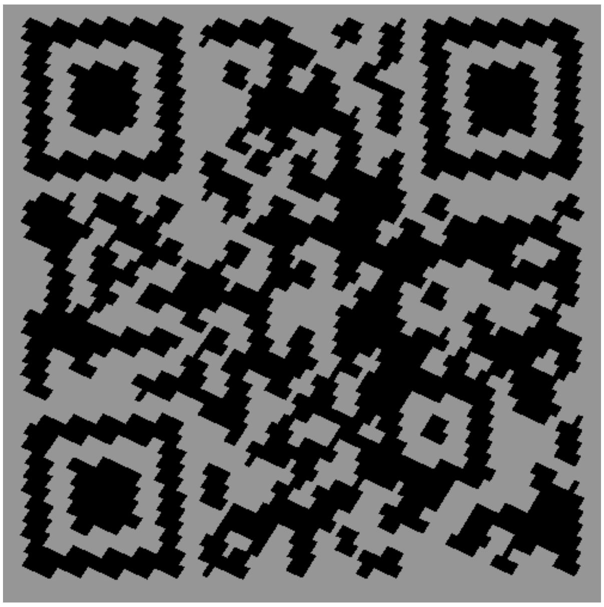
```
def warping(src, dst, H, ymin, ymax, xmin, xmax, direction='b'):
h_src, w_src, ch = src.shape
    h_dst, w_dst, ch = dst.shape
    H_inv = np.linalg.inv(H)

    # TODO: 1.meshgrid the (x,y) coordinate pairs
    x, y = np.meshgrid(np.arange(xmin, xmax), np.arange(ymin, ymax))
    # TODO: 2.reshape the destination pixels as N x 3 homogeneous coordinate
    target_pixels = np.vstack((x.flatten(), y.flatten(), np.ones(x.size)))

    if direction == 'b':
        # TODO: 3.apply H_inv to the destination pixels and retrieve (u,v) pixels, then
reshape to (ymax-ymin),(xmax-xmin)
        src_pixels = np.dot(H_inv, target_pixels)
        src_pixels = (src_pixels / src_pixels[2])[:2]
        # TODO: 4.calculate the mask of the transformed coordinate (should not exceed the
boundaries of source image)
        mask = (src_pixels[0] >= 0) & (src_pixels[0] < w_src) & (src_pixels[1] >= 0) &
(src_pixels[1] < h_src)
        # TODO: 5.sample the source image with the masked and reshaped transformed
coordinates
        valid_src_pixels_x = src_pixels[0][mask]
        valid_src_pixels_y = src_pixels[1][mask]
        valid_src_pixels = np.vstack((valid_src_pixels_x, valid_src_pixels_y))
        valid_target_pixels_x = target_pixels[0][mask]
        valid_target_pixels_y = target_pixels[1][mask]
        valid_target_pixels = np.vstack((valid_target_pixels_x, valid_target_pixels_y))
        # TODO: 6. assign to destination image with proper masking
        dst[valid_target_pixels[1].astype(np.int), valid_target_pixels[0].astype(np.int)] = src[valid_src_pixels[1].astype(np.int), valid_src_pixels[0].astype(np.int)]

    elif direction == 'f':
        # TODO: 3.apply H to the source pixels and retrieve (u,v) pixels, then reshape to (ymax-ymin),(xmax-xmin)
        dst_pixels = np.dot(H, target_pixels)
        dst_pixels = (dst_pixels / dst_pixels[2])[:2]
        # TODO: 4.calculate the mask of the transformed coordinate (should not exceed the boundaries of destination image)
        mask = (dst_pixels[0] >= 0) & (dst_pixels[0] < w_dst) & (dst_pixels[1] >= 0) & (dst_pixels[1] < h_dst)
        # TODO: 5.filter the valid coordinates using previous obtained mask
        valid_dst_pixels_x = dst_pixels[0][mask]
        valid_dst_pixels_y = dst_pixels[1][mask]
        valid_dst_pixels = np.vstack((valid_dst_pixels_x, valid_dst_pixels_y))
        valid_target_pixels_x = target_pixels[0][mask]
        valid_target_pixels_y = target_pixels[1][mask]
        valid_target_pixels = np.vstack((valid_target_pixels_x, valid_target_pixels_y))
        # TODO: 6. assign to destination image using advanced array indicing
        dst[valid_dst_pixels[1].astype(np.int), valid_dst_pixels[0].astype(np.int)] = src[valid_target_pixels[1].astype(np.int), valid_target_pixels[0].astype(np.int)]

    return dst
```

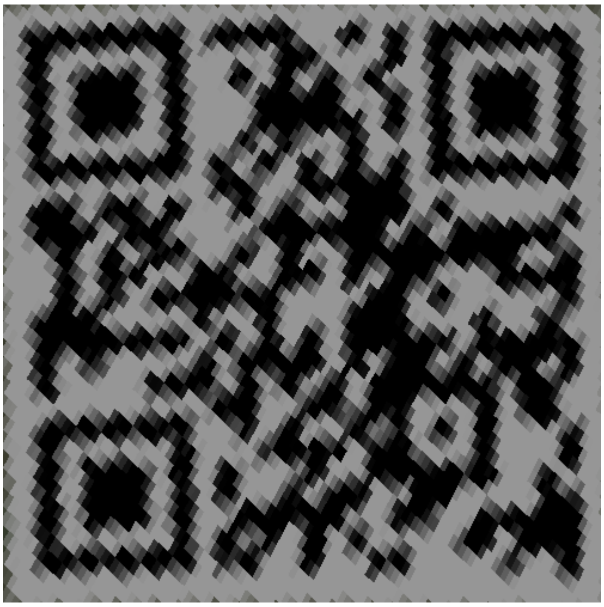
- **Briefly introduce the interpolation method you use**

當轉換後的座標不為整數時，用 numpy 的 astype 轉換回整數，astype 這個 function 會把小數部分無條件捨去。

Part 3.

- **Paste the 2 warped images and the link you find**





link: media.ee.ntu.edu.tw/courses/cv/21S/

- **Discuss the difference between 2 source images, are the warped results the same or different?**

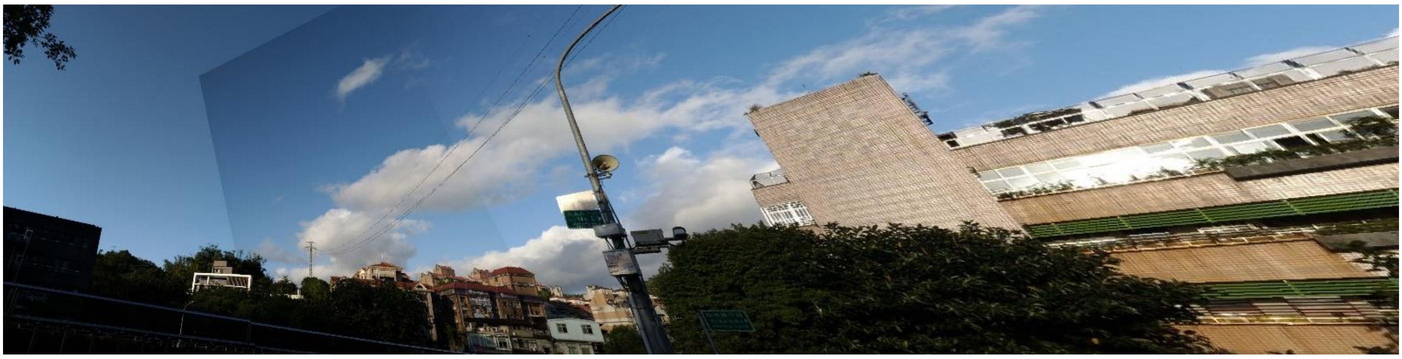
第一張 source image 的平行線仍大概維持互相平行，幾乎只有 affine transformation 的效果，而第二張 source image 則扭曲較多，較接近 projective transformation 的樣子，不過最後 warping 的結果都能掃出相同的網站。

- **If the results are the same, explain why. If the results are different, explain why?**

雖然第二張圖片產生的 qrcode 明顯比較模糊，但因為 homography 解出的變換包含 8 dof，所以能將 projection 過後的影像恢復至能顯示 qrcode 的狀態。

Part 4.

- **Paste your stitched panorama**



- **Can all consecutive images be stitched into a panorama?**

No

- **If yes, explain your reason. If not, explain under what conditions will result in a failure?**

如果拍攝的鏡頭有經過平移，讓影像不在一個平面上，那麼 feature points 就無法對應，應該就不能拼接，或是在邊界處的 feature points 不夠多時，就可能找不到對應的 homography，導致無法拼接。