EN2550 Assignment 02

180616T P.M.P.H.Somarathne

1 2D Transformations

Rotation, translation, scaling, reflection, shear are the applied as basic transformations.[1] Euclidian, similarity and affinity transformations are derived from these basic transformations. Perspective transformation is applied using a homography matrix.

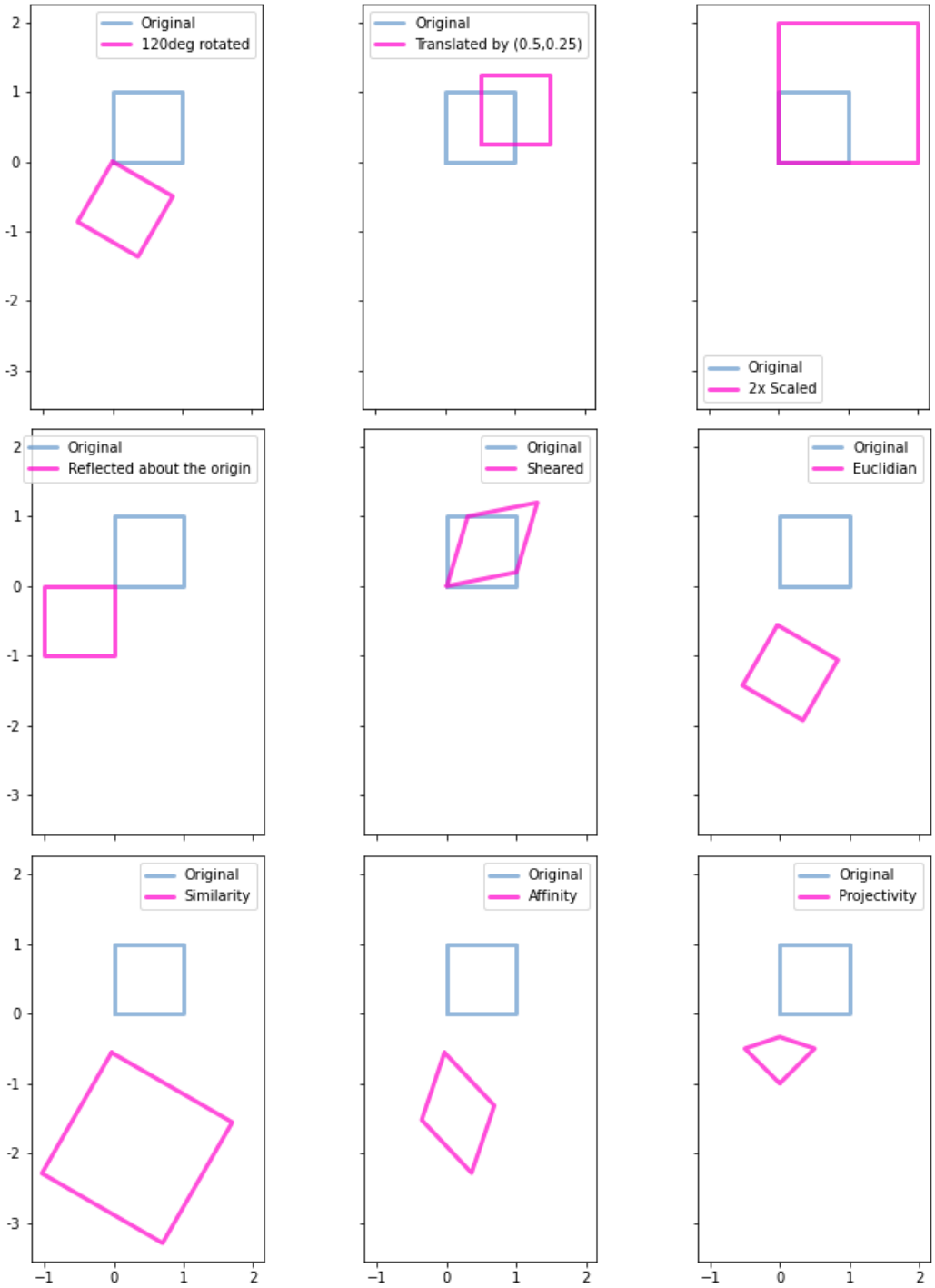


Figure 1: *2D transformation results*

2 Warping Using a Given Homography

Combined img1.ppm and img5.ppm by using the provided homography matrix(H1to5p).[2]

(a) *img1.ppm* (b) *img5.ppm* (c) *Warped and stitched image*

Figure 2: Image warping using given homography

3 Warping Using Mouse-Clicked Points and OpenCV

Combined img1.ppm and img5.ppm with homography matrix generated by mouse-clicked points and **cv2.findHomogra** function. The stitching is not as accurate as the above one due to the inaccuracies in clicking the points by the user.

Listing 1: Calculating homography using OpenCV and stitching

```
1 H, _ = cv.findHomography(p2,p1)
2 im5_warped = cv.warpPerspective(im5, H, (1000,1000))
3 im5_warped[0:im1.shape[0], 0:im1.shape[1]] = im1
```

(a) *img1.ppm* (b) *img5.ppm* (c) *Warped and stitched image*

Figure 3: Image warping mouse clicked points

4 Warping Using Mouse-Clicked Points without OpenCV

Combined img1.ppm and img5.ppm with homography matrix generated by mouse-clicked points and a function[4]. Here also stitching is not as accurate, due to the inaccuracies in clicking the points by the user.

The homography matrix(H) *corresponds to the eigenvector of* A *with smallest eigenvalue, in* $AH = 0$

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1^{'}x_1 & -x_1^{'}y_1 & -x_1^{'} \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y_1^{'}x_1 & -y_1^{'}y_1 & -y_1^{'} \\ & & & & & & & & \\ x_n & y_n & 1 & 0 & 0 & 0 & -x_n^{'}x_n & -x_n^{'}y_n & -x_n^{'} \\ 0 & 0 & 0 & x_n & y_n & 1 & -y_n^{'}x_n & -y_n^{'}y_n & -y_n^{'} \end{bmatrix} \begin{bmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \\ h_{22} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

where x_i, y_i are the coordinates of source image and $x_i^{'}, y_i^{'}$ are the corresponding destination points.

Listing 2: Calculating homography without OpenCV

```
def findHomography(srcPnts, dstPnts):
    A = []
    for i in range(len(srcPnts)):
        A.append([srcPnts[i][0], srcPnts[i][1], 1, 0, 0, 0,\
                -dstPnts[i][0]*srcPnts[i][0], -dstPnts[i][0]*srcPnts[i][1],\
                -dstPnts[i][0]])
        A.append([0, 0, 0, srcPnts[i][0], srcPnts[i][1], 1,\
                -dstPnts[i][1]*srcPnts[i][0], -dstPnts[i][1]*srcPnts[i][1],\
                -dstPnts[i][1]])
    A = np.array(A)
    eigval, eigvect = np.linalg.eig(A.T@A)
    ind = eigval.argsort()[0]
    return eigvect.T[ind].reshape((3,3))
```

(a) *img1.ppm* (b) *img5.ppm* (c) *Warped and stitched image*

Figure 4: Image warping mouse clicked points without OpenCV

5 BONUS

5.1 Computing the Homography Using SuperGlue and RANSAC

Used SuperGlue Pretrained Network[3] to generate the matching pairs. From the pairs, only using pairs with above 90% confidence. Homography matrix is generated by **cv2.findHomography** with the method as **cv2.RANSAC**. RANSAC re-projection threshold is given as 5.0

Listing 3: Using points from SuperGlue for homography generation

```
# Accessing the results
npz = np.load('graffiti\output\img1_img5_matches.npz')
keypoints0,keypoints1 = npz['keypoints0'],npz['keypoints1']
pairs,confidence = npz['matches'],npz['match_confidence']
```

```
p1,p2 = [],[]
for i in range(keypoints0.shape[0]):
    if pairs[i]!=-1 and confidence[i]>0.75:
        p1.append(keypoints0[i])
        p2.append(keypoints1[pairs[i]])
p1,p2 = np.array(p1),np.array(p2)
# Using RANSAC to calculate the homography
H,_ = cv.findHomography(p2,p1,cv.RANSAC,5.0)
```

(a) *SuperGlue Output*

(b) *Image warped with SuperGlue and RANSAC*

Figure 5: Image warping using SuperGlue and RANSAC

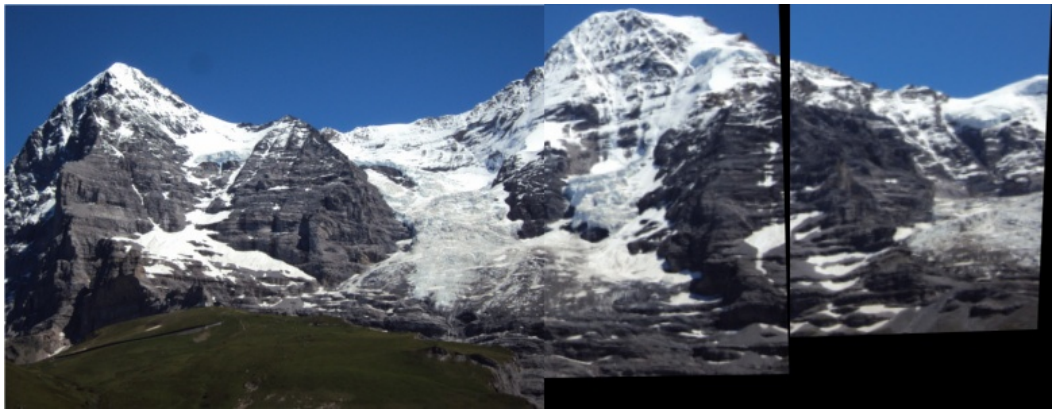
5.2 Stitching multiple images

Continuously applied mouse-clicked point extraction, homography calculation with OpenCV and stitching with OpenCV to stitch 3 images of the hill. After each stitching the black strips at the right edge and bottom edge have been removed to allow next stitching process. However, the seam handling approach could not be completed accurately.

(a) *Hill 1*

(b) *Hill 2*

(c) *Hill 3*



(d) *Stitched panorama image*

Figure 6: Panorama stitching process

5.3 Replacement for cv2.warpPerspective

I have tried to code a function for stitching images similar to **cv2.warpPerspective**. However, I couldn't debug it completely. Below is the code and the resultant image from the current revision.

Listing 4: warpPerspective test function

```
def warpPerspective(image, homography):
    width,height,channels = image.shape
    a, b, c, d = (0,0,1), (0,height-1,1), (width-1,height-1,1), (width-1,0,1)
    corners = np.array([a, b, c, d]).T
    newCorners = np.matmul(homography,corners)
    newCorners = (newCorners/newCorners[2,:]).astype(int)
    maxSize = int(np.max(newCorners))
    warpedImage = np.zeros((maxSize,maxSize,channels),dtype=np.uint8)
    h_inv = np.linalg.inv(homography)
    for i in range(maxSize):
        for j in range(maxSize):
            oldCoord = np.matmul(h_inv, [i,j,1])
            oldCoord = (oldCoord/oldCoord[2]).astype(int)
            x,y = oldCoord[:2]
            if (0<=x<width) and (0<=y<height): warpedImage[i,j] = image[x,y]
    return warpedImage
```

(a) *img1.ppm* (b) *img4.ppm* (c) *Output of the function*

Figure 7: warpPerspective test function

Full code available at `https://github.com/PamudithaSomarathne/EN2550/tree/master/1%20Alignment`

References

[1] 2D-Translation in Computer Graphics
`https://www.gatevidyalay.com/2d-transformation-in-computer-graphics-translation-examples/`

[2] Graffiti Image Set
`https://www.robots.ox.ac.uk/~vgg/data/affine/`

[3] SuperGlue Inference and Evaluation Demo Script
`https://github.com/magicleap/SuperGluePretrainedNetwork`

[4] Lecture 13: Homographies and RANSAC, MIT CSAIL
`http://6.869.csail.mit.edu/fa12/lectures/lecture13ransac/lecture13ransac.pdf`