BLG 453E Homework - 3

Image Morphing

Musa Anıl Doğan

150130053

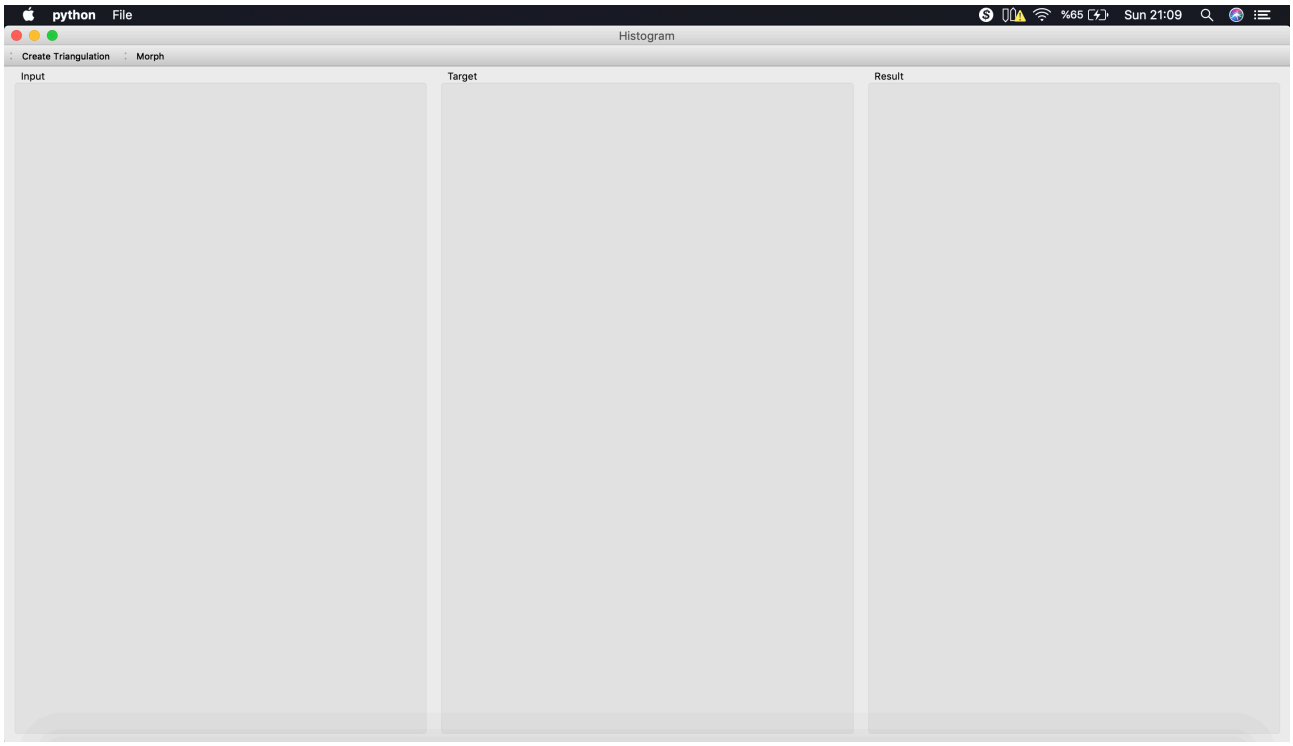
09.12.2018

Introduction

First of all, in this project, should understand how to find affine matrix and do backward mapping by using affine matrix so I use these equation;
[x y 1]= A x [x' y' 1]

Interface

Project's interface is like that;



1. Making an interface for showing Input, Target and Result images

Using Widget function to create widget then create QGroupBox widget to create 1 box for interface(Input).

In the GroupBox, layout should be set because layout is needed that for upload image and plots to the interface.

Finally, set these widget to the main widget

2. Creating menu bar

Using menuBar() function to create file bar and then create also toolbar. Toolbar has 2 actions; one of them is create triangulation, one of them is Morph. If we press Create Triangulation It provides us to create Delaunay triangles for input and target Images.
If we press Morph, it provides us to do warping these two images.

3. Create delaunay triangles

First need to choose all keypoints for Input and Target images, then put them to subdiv2D function and then draw delaunay triangles. Drawing delaunay triangles function is like that;

```
def draw_delaunay(self, subdiv ) :
    triangleList = subdiv.getTriangleList();
    size = self.image.shape
    r = (0, 0, size[1], size[0])
    for t in triangleList :
        pt1 = (t[0], t[1])
        pt2 = (t[2], t[3])
        pt3 = (t[4], t[5])
        if self.rectcontains(r, pt1) and self.rectcontains(r, pt2) and self.rectcontains(r, pt3) :
            cv2.line(self.triimage, pt1, pt2, self.delaunay_color, 1, cv2.LINE_AA, 0)
            cv2.line(self.triimage, pt2, pt3, self.delaunay_color, 1, cv2.LINE_AA, 0)
            cv2.line(self.triimage, pt3, pt1, self.delaunay_color, 1, cv2.LINE_AA, 0)

    self.qImg = QImage(self.triimage.data,size[1],size[0],size[1]*3,QImage.Format_RGB888).rgbSwapped()

    self.imageLabel.setPixmap(QPixmap.fromImage(self.qImg))
    self.imageLabel.setAlignment(Qt.AlignCenter)
    self.inputBox.layout().addWidget(self.imageLabel)
```

4. Morphing

First we need to find affine matrix and then do backward mapping. After all, final thing that we should do is interpolation. I use nearest interpolation. My code is like that;

Finding Affine Matrix;

```
bmatrix1=[a[1],c[1],d[1]] # target image x's
mmatrix = [[ptt1[1],ptt1[0],1],[ptt2[1],ptt2[0],1],[ptt3[1],ptt3[0],1]]# input image
bmatrix2 =[a[0],c[0],d[0]] # target image y's
minverse = inv(mmatrix)
amatrix1 = np.matmul(minverse,bmatrix1)
amatrix2 = np.matmul(minverse,bmatrix2)
amatrix = [[amatrix1[0],amatrix1[1],amatrix1[2]],[amatrix2[0],amatrix2[1],amatrix2[2]],[0,0,1]]
```

Backward mapping;

```
for a in range(0,heightI):
    for b in range(0,widthI):
        bx=targettriangle[1][1]-targettriangle[0][1]
        by=targettriangle[1][0]-targettriangle[0][0]
        cx=targettriangle[2][1]-targettriangle[0][1]
        cy=targettriangle[2][0]-targettriangle[0][0]
        x=a-targettriangle[0][1]
        y=b-targettriangle[0][0]
        d=bx*cy-cx*by
        wa = (x*(by-cy)+y*(cx-bx)+bx*cy-cx*by)/d
        wb = (x*cy-y*cx)/d
        wc = (y*bx-x*by)/d
        if wa>0 and wa<1 and wb>0 and wb<1 and wc>0 and wc<1:
            coord = np.matmul(invamatrix,[a,b,1])
            coordinate = (coord[0],coord[1])
            coordt.append(coordinate)
```

Interpolation;

```
if coordt[index][1]>=widthI:
    newy=int(coordt[index][1])-320
    #coordt[index][1]=coordt[index][1]-320
else:
    newy =int(coordt[index][1])
if coordt[index][0]>=heightI:
    newx=int(coordt[index][0])-480
    #coordt[index][0]=coordt[index][0]-480
else:
    newx=int(coordt[index][0])
#self.image[a][b]=self.image[newx][newy]
#self.image[a][b]=self.getbilinearpixel(newx,newy)
self.result[a][b] = self.image[newx][newy]
index=index+1
```

Conclusion

The development of my skills such as "python", "opencv", "image processing" and "PYQT5" was very nice for me. At the same time, this project is really useful for understanding mathematics of affine matrix transformation and interpolation.